UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE MÚSICA E ARTES CÊNICAS

SÉRGIO DE ALENCASTRO VEIGA FILHO

MÚSICA ELETROACÚSTICA UTILIZANDO *SOFTWARE* LIVRE: PROCESSOS COMPOSICIONAIS INTERATIVOS

Goiânia
2015

sistema de bibliotecas ufg

UFG

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR AS TESES E DISSERTAÇÕES ELETRÔNICAS (TEDE) NA BIBLIOTECA DIGITAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio da Biblioteca Digital de Teses e Dissertações (BDTD/UFG), sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1. Identificação do material bibliográfico: **[X] Dissertação** **[ ] Tese**

2. Identificação da Tese ou Dissertação

Autor (a):	Sérgio de Alencastro Veiga Filho				
E-mail:	sergim.veiga@gmail.com				
Seu e-mail pode ser disponibilizado na página?	[X]Sim	[ ] Não			
Vínculo empregatício do autor	Técnico em Audiovisual				
Agência de fomento:	Fundação de Amparo à Pesquisa do Estado de Goiás		Sigla:	FAPEG	
País:	Brasil	UF:GO	CNPJ:	01567601/0001-43	
Título:	Música Eletroacústica Utilizando Software Livre: Processos Composicionais Interativos				
Palavras-chave:	Música Eletroacústica, Interface Interativa, Software Livre				
Título em outra língua:	Electroacoustic Music Using Free Software: Interactive Compositional Processes				
Palavras-chave em outra língua:	Electroacoustic Music, Interactive Interface, Free Software				
Área de concentração:	Música na Contemporaneidade				
Data defesa: (dd/mm/aaaa)	27/04/2015				
Programa de Pós-Graduação:	Mestrado em Música				
Orientador (a):	Anselmo Guerra de Almeida				
E-mail:	guerra.anselmo@gmail.com				
Co-orientador (a):*					
E-mail:					

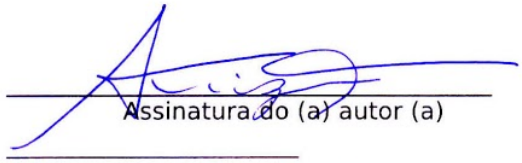
*Necessita do CPF quando não constar no SisPG

3. Informações de acesso ao documento:

Concorda com a liberação total do documento [X] SIM [ ] NÃO[1]

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do(s) arquivo(s) em formato digital PDF ou DOC da tese ou dissertação.

O sistema da Biblioteca Digital de Teses e Dissertações garante aos autores, que os arquivos contendo eletronicamente as teses e ou dissertações, antes de sua disponibilização, receberão procedimentos de segurança, criptografia (para não permitir cópia e extração de conteúdo, permitindo apenas impressão fraca) usando o padrão do Acrobat.



Assinatura do (a) autor (a)

Data: 25/05/2015

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.

SÉRGIO DE ALENCASTRO VEIGA FILHO

MÚSICA ELETROACÚSTICA UTILIZANDO *SOFTWARE* LIVRE:
PROCESSOS COMPOSICIONAIS INTERATIVOS

Trabalho final (produção artística e artigo) apresentado ao Programa de Pós-Graduação Stricto Sensu da Escola de Música e Artes Cênicas da Universidade Federal de Goiás, como requisito parcial para a obtenção do título de Mestre em Música.
Área de Concentração: Música na Contemporaneidade
Linha de Pesquisa: Música, Criação e Expressão
Orientador: Prof. Dr. Anselmo Guerra de Almeida.

Goiânia
2015

Ficha catalográfica elaborada automaticamente
com os dados fornecidos pelo(a) autor(a), sob orientação do Sibi/UFG.

Veiga Filho, Sérgio de Alencastro
Música Eletroacústica Utilizando Software Livre: [manuscrito] : Processos Composicionais Interativos / Sérgio de Alencastro Veiga Filho. - 2015.
x, 86 f.

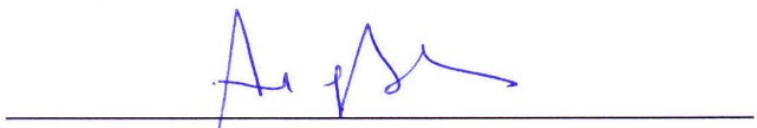
Orientador: Prof. Dr. Anselmo Guerra de Almeida.
Dissertação (Mestrado) - Universidade Federal de Goiás, Escola de Música e Artes Cênicas (Emac) , Programa de Pós-Graduação em Música, Goiânia, 2015.
Bibliografia. Anexos.
Inclui siglas, algoritmos, lista de figuras.

1. Música Eletroacústica. 2. Interface Interativa. 3. Software Livre. I. Almeida, Anselmo Guerra de, orient. II. Título.

SÉRGIO DE ALENCASTRO VEIGA FILHO

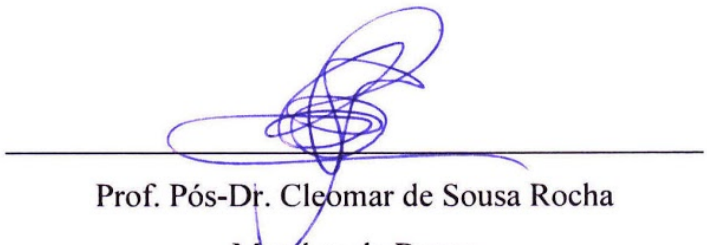
"Música Eletroacústica Utilizando *Software* Livre:
Processos Composicionais Interativos"

Trabalho final de curso defendido e aprovado em 27 de Abril de 2015, perante a Banca Examinadora constituída pelos professores:



Prof. Dr. Anselmo Guerra de Almeida
Orientador e Presidente da Banca

Prof. Dr. Carlos Henrique Costa
Membro da Banca



Prof. Pós-Dr. Cleomar de Sousa Rocha
Membro da Banca

Agradeço primeiramente à minha família pelo total incentivo em minha carreira e pela exposição à boa harmonia desde a infância. Agradeço profundamente à minha esposa pelo carinho, amor, revisão deste texto e escrita japonesa no computador e à mão.

AGRADECIMENTOS

Agradeço imensamente à orientação por mais de 10 anos pelo amigo compositor Anselmo Guerra de Almeida mesmo antes de ingressar ao mestrado.

Agradeço muito ao Professor Dr. Cleomar de Sousa Rocha pelas orientações na formatação e linguagem. Ao Professor Dr. Fábio Oliveira por orientar questões estéticas e artísticas em minha qualificação.

À Professora Drª Sônia Ray pelas ações preventivas, ao Professor Dr. Carlos Henrique Costa pela sensibilidade e a todos os colegas de trabalho dentro da escola de música da UFG que considero todos amigos.

Aos amigos compositores que conheci ao longo dessa jornada e tanto enriqueceram meu conhecimento musical na minha graduação como Estércio Marquez Cunha, Conrado Silva de Marco, Jorge Antunes, Flávio Santos Pereira, Paulo Guicheney, Christiano Figueiró e aos demais alunos do bacharelado em composição atualmente.

Ao meu irmão Samuel por todo conhecimento que me deu em processamento de dados e que possibilitou este trabalho associativo de música e informática.

À FAPEG, por me conceder uma bolsa de estudos durante o mestrado, possibilitando meu aprofundamento na pesquisa musical e aquisição de livros e equipamentos tecnológicos para esta pesquisa.

Ao LPqS/EMAC-UFG, que acolheu o projeto onde pudemos realizar toda pesquisa experimental, ampliar o nosso conhecimento tecnológico e pesquisa interdisciplinar.

RESUMO

Este é um trabalho que investiga a possibilidade de composição de obras musicais com interfaces interativas criadas com algoritmos e programas que sejam *Softwares* Livres desde o sistema operacional. Inclui também, descrição de suas elaborações, tanto para as composições musicais com interfaces já existentes como para as interfaces construídas à luz da formação musical e tecnológica deste compositor. As composições foram apresentadas durante o curso de mestrado em apresentações artísticas nos SEMPEM XIII, SEMPEM XIV e apresentações de qualificação e defesa. Este trabalho de pesquisa portanto se divide em duas partes: a Parte A, que consiste da parte artística onde o compositor apresenta sua obra para audição pública, e Parte B, que consiste em todo trabalho escrito, das partituras e algorítmos em linguagem computacional ao presente artigo.

Palavras chave: Música Eletroacústica, Interface Interativa, Software Livre.

ABSTRACT

This is a work investigating the possibility of composing musical works with interactive interfaces created with algorithms and programs that are Free Software including the entire operating system. It also includes description of their elaborations, both musical compositions with existing interfaces as for interfaces built in the light of musical and technological training of this composer. The compositions were presented during the Master's degree in performing arts in SEMPEM XIII, XIV and SEMPEM qualification and defense presentations. This research therefore is divided into two parts: Part A, which consists of the artistic part where the composer presents his work to the public hearing, and Part B, which consists of all written work, the scores and algorithms in computer language to this article.

Keywords: Electroacoustic Music, Interactive Interface, Free Software.

LISTA DE FIGURAS

SUMÁRIO

PARTE A: PRODUÇÃO ARTÍSTICA

UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
XIII SEMPEM
04 de Outubro de 2013 – 09:15h
1ª Apresentação

XIII SEMPEM - RECITAL 6

PROGRAMA

RABBATH, François (1931) — Ode D'Espagne (1963)

COLTRANE, John (1926 -1967) — Mr. Pc (1959)

Gadiego Carraro, Contrabaixo acústico solo

SEDÍCIAS, Dimas (1930 - 2001) (1986) — Mutreta para duas Clarinetas

MURCIA, Mauricio (1976 -) — Mauro's Latin (2006)

LICHTENZVEIG, Osvaldo (1976 -) — La Minon-ga (2010)

Rosa Barros e **Taís Vilar**, Clarinetas

CUNHA, Estércio Marquez (1941) (2013) — Música para Trombone N. 4

I – movimento: Lento

Jackes Douglas, Trombone

VILLANI-CÔRTES, Edmundo (1930) — Oito peças para piano

Casulo (1990)
Baile imaginário (1989)
Valsinha de roda (1979)
O passarinho da praça da matriz (1994)
A fonte eterna (1974)
Balada dos 15 minutos (1987)
Canção de Carolina (1990)
Sem nome (2012)

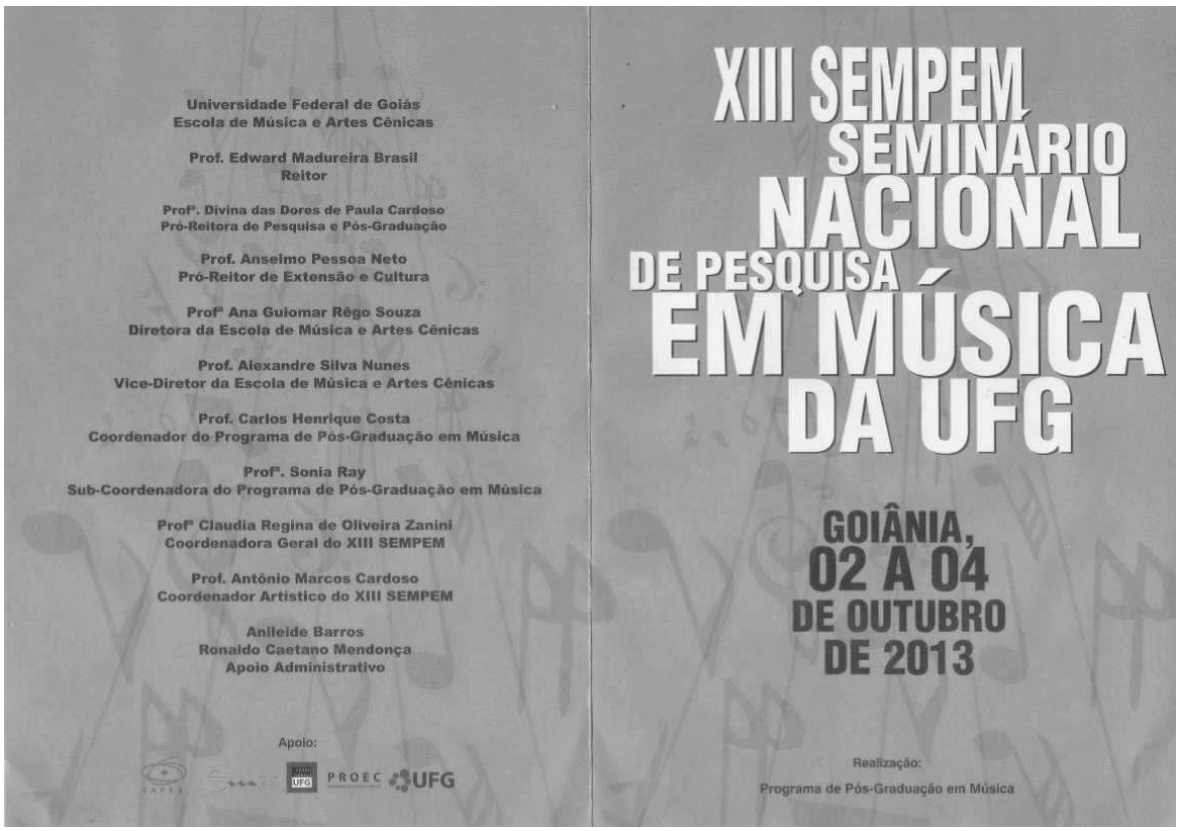
Rúbia Santos, piano

VEIGA FILHO, Sérgio de Alencastro (1977) — Música para ThereminGiroscópico e Granulações Harmônicas (2013)

I – Vênus
II – Marte

Sergio Veiga, Interface interativa (Theremin-Giroscópico)

TEATRO DA EMAC
04/10/2013
9:15hs



UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
XIV SEMPEM
01 de Outubro de 2014 – 16:00h
2ª Apresentação

TrioBSB Formado por dois solistas de orquestra sinfônica com grande experiência e uma especialista em música de câmera ao piano, o Trio BSB dedica-se à música de câmara moderna brasileira, bem como dos compositores universais. Ao se unir o fraseado da flauta, aliado à voz do violoncelo e à base firme do piano, o resultado inevitável são peças que soam por meio de grandes instrumentistas trazendo brilho ao cenário onde o grupo toca. O TrioBSB tem se apresentado regularmente, e também se dedica ao trabalho de gravação em estúdio, cujo objetivo é permitir a criação de um legado para maior divulgação dos compositores nacionais e venha a trazer impacto também no meio da educação musical no país. **Flauta – José Evangelista Violoncelo – Rodolpho Borges Piano – Gisele Pires-Mota**

O **Duo Limiares**, flauta Sara Lima e piano Robervaldo Linhares, com a participação do violinista Luciano Pontes, interpreta três obras do jovem compositor Luiz Gonçalves (1986). Este grupo, formado por músicos e estudante de composição ligados à EMAC-UFG, apresenta o "Pequeno Concerto Zen". A primeira obra é uma peça para piano solo, Cinco Prelúdios para Meditação, dedicada ao pianista Robervaldo Linhares. A segunda, Variações sobre um leque, é um delicado duo de flauta e piano com referências à música oriental. A última é um trio de flauta, violino e piano: Farewell y los Sollozos. Trata-se de uma composição expressiva e pungente que sugere a dor da despedida, seus possíveis soluços e sua possível superação.

PROGRAMA

RECITAL 6

GONÇALVES, Luiz (1986) Prelúdios para meditação II e IV* (2011) estreia mundial
GONÇALVES, Luiz (1986) Variações sobre um leque** (2007)
GONÇALVES, Luiz (1986) Farewell y los Sollozos*** (2013)
I - Con moto
II – Etéreo
III – Moderato

* piano solo
** flauta e piano
*** violino, flauta e piano

Duo Limiares

MIGNONE, Francisco (1897-1986) Trio para flauta, piano e cello n. 1 (1981)

I. Andante
II. Modinha
III. Festança sem boi

TrioBSB

CONCERTO DE MÚSICA ELETROACÚSTICA

VEIGA, Sérgio (1977) Caverna da medusa (2014)

Improviso

Alunos do Curso de Live Electronics
Professor Alexandre Torres Porres (USP)
Curadoria: Anselmo Guerra (UFG)

01/10/2014
16:00
TEATRO DA EMAC

UNIVERDADE FEDERAL DE GOIÁS

Prof. Orlando Afonso Valle do Amaral
Reitor da UFG

Prof º José Alexandre Felizola Diniz Filho
Pró-reitor de Pós-Graduação

Prof ª Maria Clorinda Soares Fiarovanti
Pró-reitora de Pesquisa e Inovação

Prof ª Giselle Ferreira Ottoni Candido
Pró-reitora de Extensão e Cultura

Profa. Ana Guiomar Rêgo Souza
Diretora da EMAC/UFG

Prof. Carlos Henrique Costa
Coordenador do PPG-Música
Presidente da Comissão Organizadora do SEMPEM

Prof. Antônio Marcos Cardoso
Coordenador da Comissão Artística do SEMPEM

Prof. Werner Aguiar
Coordenador da Comissão Científica do SEMPEM

APOIO

emac PROEC UFG
centro cultural UFG FAPEG GOVERNO DE GOIÁS

UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
08 de Dezembro de 2014 – 9:30h
Apresentação de Qualificação

SERGIM VEIGA (1977-)

Vênus e Marte (2013)

Para Theremin Giroscópico e Síntese por Randomização de Harmônicos.

1ª Parte: Vênus

2ª Parte: Marte

Caverna da Medusa (2014)

Para duas interfaces do compositor e quatro interfaces de Iain McCurdy.

Saigai to Higai 災害と被害 (2014)

Para acelerômetro de qualquer dispositivo celular.

Counter's Jazz (2014)

Para dois *sliders:* Transposição e Transfiguração.

Recital apresentado ao Programa de Pós-Graduação em Música da EMAC UFG como requisito parcial para a obtenção do título de Mestre em Música.
Banca: Prof. Dr. Anselmo Guerra de Almeida
Prof. Pós-Dr. Cleomar de Sousa Rocha
Prof. Dr. Fábio Fonseca de Oliveira

UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
08 de Dezembro de 2014 – 9:30h
Apresentação de Qualificação

Notas de Programa

Sérgio de Alencastro Veiga Filho é Bacharel em Composição Musical pela Universidade Federal de Goiás (2003). Desde 2004 realiza pesquisas obre harmonia musical em contextos micro-tonais e macro-tonais com a utilização de síntese sonora computacional. À partir de 2009 inicia pesquisas com interfaces interativas. Hoje cursa o Mestrado em Música do Programa de Pós Graduação da UFG. Fomentado pela bolsa da FAPEG.

Sobre as Composições:

VÊNUS E MARTE

A motivação poética da composição de Vênus e Marte começou no desenvolvimento de interfaces interativas, que de fato, são composições algorítmicas em linguagem computacional. A interface Sintetizador por Randomização de Harmônicos é o instrumento harmônico da música, o elemento que relacionamos poeticamente ao masculino, gerador de milhões de sementes, Marte o "Deus da Guerra". O Theremin Giroscópico foi associado ao lado feminino, dentro dessa intenção poética, e à qual relacionamos a Vênus. É muito similar ao Theremin original, muitas vezes lembra voz de mulher e mesmo em regiões mais graves, ela é o cântico melódico acompanhado com harmônicos cordais. Podemos considerar o princípio básico de Vênus e Marte como sendo duas canções com uma peculiar textura de melodia acompanhada que na verdade é à duas vozes, onde Marte, acompanha Vênus com notas sucessivas.

CAVERNA DA MEDUSA

Caverna da Medusa foi composta à partir de 2 (duas) interfaces criadas durante estudos de Csound e 4 (quatro) interfaces criadas por Iain McCurdy que estão disponíveis em seu site. O tema não inspirou a música, foi o inverso, a música, depois de quase pronta, inspirou o nome. Ela evocava lembranças de histórias sobre mitologia grega porque os efeitos de vozes da interface "FoFx6" quando congelam no tempo remetia às estátuas das pessoas que transformam em pedra quando encaram a Medusa. O som do "Delay Doppler", que lembra vento encanado, foi adicionado depois da idéia do nome, para trazer a sensação do mundo subterrâneo onde se passa o mito.

SAIGAI TO HIGAI 災害と被害

De acordo com Coelho e Hida (1999), Saigai significa, a calamidade, a catástrofe, o desastre, o sinistro, o acidente natural. Higai significa, o dano, o prejuízo, o estrago causado pelo homem. A composição é uma narrativa abstrata sobre os eventos ocorridos no Japão em março de 2011 e se divide em 3 partes: o Terremoto (Jishin-地震), o Maremoto (Tsunami-津波) e o Vazamento Radioativo (Hōsha no More-放射の漏れ).

COUNTER'S JAZZ

Inspirado nas progressões harmônicas por ciclo de quintas e cromatismos com intervalo de trítono encontrados no jazz, essa composição faz uso de contadores nos *loops* de cada linha melódica para dar prosseguimento a essas progressões lógicas. Constitui-se de parte A e B e quando retoma do início, começa o improviso do intérprete que pode alterar 2 parâmetros: a transposição e o redimensionamento intervalar.

UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
08 de Dezembro de 2014 – 9:30h
Apresentação de Qualificação

Capa de Programa

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE MÚSICA E ARTES CÊNICAS
PROGRAMA DE PÓS-GRADUAÇÃO EM MÚSICA

SÉRGIO DE ALENCASTRO VEIGA FILHO
Compositor

APRESENTAÇÃO DE QUALIFICAÇÃO

Teatro Belkiss Spenzieri Mendonça
Goiânia, Segunda-Feira, 08 de Dezembro de 2014
Horário 09:30

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE MÚSICA E ARTES CÊNICAS

Profa. Dra. Ana Guiomar Rêgo Souza
Diretora da Escola de Música e Artes Cênicas

Prof. Dr. Carlos Henrique Costa
Coordenador do Programa de Pós-Graduação em Música

Profa. Dra. Sonia Ray
Vice-Coordenadora do Programa de Pós-Graduação em Música

UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
27 de Abril de 2015 – 10:00h
Apresentação de Defesa

SERGIM VEIGA (1977-)

Vênus e Marte (2013)

Para Theremin Giroscópico e Síntese por Randomização de Harmônicos.

1ª Parte: Vênus

2ª Parte: Marte

Caverna da Medusa (2014)

Para duas interfaces do compositor e quatro interfaces de Iain McCurdy.

Saigai to Higai 災害と被害 (2014)

Para acelerômetro de qualquer dispositivo celular.

ANS Estudo Nº1 (2015)

Modelagem espectral em VirtualANS.

ANS Estudo Nº2 (2015)

Modelagem espectral em VirtualANS.

Recital apresentado ao Programa de Pós-Graduação em Música da EMAC UFG
como requisito parcial para a obtenção do título de Mestre em Música.
Banca: Prof. Dr. Anselmo Guerra de Almeida
Prof. Dr. Carlos Henrique Costa
Prof. Pós-Dr. Cleomar de Sousa Rocha

UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
27 de Abril de 2015 – 10:00h
Apresentação de Defesa

Notas de Programa

Sérgio de Alencastro Veiga Filho é Bacharel em Composição Musical pela Universidade Federal de Goiás (2003). Desde 2004 realiza pesquisas obre harmonia musical em contextos micro-tonais e macro-tonais com a utilização de síntese sonora computacional. À partir de 2009 iniciou pesquisas com interfaces interativas. Hoje, cursa o Mestrado em Música do Programa de Pós Graduação da UFG e é fomentado por bolsa da FAPEG.

Sobre as Composições:

VÊNUS E MARTE

A motivação poética da composição de Vênus e Marte começou no desenvolvimento de interfaces interativas, que de fato, são composições algorítmicas em linguagem computacional. A interface *Sintetizador por Randomização de Harmônicos* é o instrumento harmônico da música, o elemento que relacionamos poeticamente ao masculino, gerador de milhões de sementes, Marte o "Deus da Guerra". O *Theremin Giroscópico* foi associado ao lado feminino, dentro dessa intenção poética, e à qual relacionamos a Vênus. O timbre é muito similar ao Theremin original, lembra voz de mulher e em regiões mais graves, ainda é o cântico melódico acompanhado por harmônicos cordais.

CAVERNA DA MEDUSA

Caverna da Medusa foi composta à partir de 2 (duas) interfaces criadas durante estudos de *Csound* e 4 (quatro) interfaces criadas por Iain McCurdy que estão disponíveis em seu site. O tema não inspirou a música, foi o inverso, a música, depois de quase pronta, inspirou o nome. Ela evocava lembranças de histórias sobre mitologia grega diante dos efeitos de vozes da interface "FoFx6", que quando passam pelo limite entre ritmo e frequência audível, remetiam às estátuas das pessoas que transformam em pedra quando olham nos olhos da Medusa.

SAIGAI TO HIGAI 災害と被害

De acordo com Coelho e Hida (1999), Saigai significa, a calamidade, a catástrofe, o desastre, o sinistro, o acidente natural. Higai significa, o dano, o prejuízo, o estrago causado pelo homem. A composição é uma narrativa abstrata sobre os eventos ocorridos no Japão em março de 2011 e se divide em 3 partes: o Terremoto (Jishin-地震), o Maremoto (Tsunami-津波) e o Vazamento Radioativo (Hōsha no More-放射の漏れ).

ANS Estudo Nº1 e Estudo Nº2

Muito recentemente, um desenvolvedor de software chamado Alexander Zolotov criou um aplicativo para celulares e tablets que é livre (grátis) em computadores. Trata-se de uma versão virtual do chamado sintetizador ANS, inciais do nome de Scriabin, homenagem ao compositor russo. Esse sintetizador foi inventado entre 1938 a 1958 por Evgeny Murzin (1914-1970). A composição do primeiro estudo do ANS se restringe à narrativa sonora por modelagem espectral feita com as ferramentas existentes no programa, logo, é um estudo sobre os resultados sonoros que podem ser obtidos pela síntese direta das possibilidades gráficas de suas ferramentas de trabalho, não envolvendo portanto, processamento de sons acústicos ou a utilização de pincéis que não sejam os padrões do programa. O segundo estudo é estruturado por dois temas: sujeito e contra-sujeito, onde o sujeito é a sonoridade obtida da imagem do Kanji 音楽 (ongaku) cujo significado é Música. O contra-sujeito é a sonoridade obtida da imagem do Kanji 愛 (ai) cujo significado é amor.

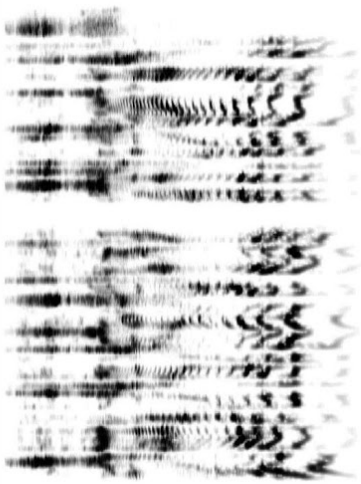
UNIVERSIDADE FEDERAL DE GOIÁS
Escola de Música e Artes Cênicas
Programa de Pós-Graduação em Música
Auditório Belkiss Spenzieri Carneiro de Mendonça
27 de Abril de 2015 – 10:00h
Apresentação de Defesa

Capa de Programa

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE MÚSICA E ARTES CÊNICAS
PROGRAMA DE PÓS-GRADUAÇÃO EM MÚSICA

SÉRGIO DE ALENCASTRO VEIGA FILHO
Compositor

APRESENTAÇÃO DE DEFESA



Teatro Belkiss Spenzieri Mendonça
Goiânia, Segunda-Feira, 27 de Abril de 2015
Horário 10:00

UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE MÚSICA E ARTES CÊNICAS

Profa. Dra. Ana Guiomar Rêgo Souza
Diretora da Escola de Música e Artes Cênicas

Prof. Dr. Carlos Henrique Costa
Coordenador do Programa de Pós-Graduação em Música

Profa. Dra. Sonia Ray
Vice-Coordenadora do Programa de Pós-Graduação em Música

PARTE B: ARTIGO

MÚSICA ELETROACÚSTICA UTILIZANDO *SOFTWARE* LIVRE: PROCESSOS COMPOSICIONAIS INTERATIVOS

INTRODUÇÃO

Apesar da polarização estética e ideológica entre a música concreta francesa com Pierre Schaeffer no *Club d'Essai* e a música eletrônica alemã com Stockhausen no estúdio de Eimert na cidade de Colônia, a Música Eletroacústica nasceu quando os compositores começaram a juntar os procedimentos das duas linhas estéticas em suas obras, e o termo "eletroacústico" foi introduzido nessa época, na década de 1950. Nesse começo, havia a busca por estruturação mais rígida do serialismo, mas com o tempo, tornou-se gênero para muitas composições musicais que misturam sons acústicos e sintéticos e/ou transformam o som por meio eletrônico na busca pelo inaudito. Introduzimos aqui um breve relato histórico focado em nosso tema: a composição musical com interfaces computacionais e relatos de processos.

O surgimento da música eletroacústica introduziu um novo meio de relação entre o compositor e o público sem a mediação do intérprete instrumentista. Os compositores estavam no início da manipulação do som por meio de síntese e gravações em fita e podiam ouvir suas obras durante ou logo após sua criação. A tentativa de interagir com uma parte fixa incomodava pela rigidez do pulso da máquina e a falta de interação dela para com os humanos. Isso culminou na tentativa de Pierre Boulez, no IRCAM (*Institut de Recherche et Coordination Acoustique/Musique*)[1], de criar um sistema em que o computador pudesse se integrar ao pulso hierárquico do intérprete, tal qual um músico que segue a partitura (*score follower*) atento ao andamento geral. Então em 1997, a composição *Anthèmes 2* aponta o estado-da-arte do *score follower*, onde Boulez atinge seu objetivo (Boulez, 2001).

Antes dessa evolução estética na música, começaram a eclodir instrumentos musicais com interfaces não convencionais no começo do século XX. O estudo da eletricidade, a invenção da válvula e o registro sonoro em película pela variação da intensidade da luz foram ferramentas no desenvolvimento de sintetizadores analógicos. Com válvulas elétricas o cientista russo Lev Sergeyevich Termen [Лев Сергеевич Термен], conhecido no ocidente como Léon Theremin descobriu o fenômeno da heterodinização. Músico, criou um instrumento musical com essa descoberta e o nomeou Etherophone, instrumento com interface impressionante para sua época, ficou famoso por ser tocado sem contato físico, apenas a variação da distância das mãos entre suas antenas são suficientes para

[1] Instituto de Pesquisa e Coordenação de Música e Acústica (http://www.ircam.fr/). Acessado em Abril de 2015.

o controle de amplitude e frequência, sendo, uma antena horizontal à esquerda para a intensidade e outra antena vertical à direita para as alturas das notas. Atualmente o instrumento é conhecido pelo nome de seu criador.

Com base em variações luminosas para o registro sonoro, o sintetizador ANS (*Alexander Nikolayevich Scriabin*)[2] foi desenvolvido na Rússia por Evgeny Murzin (1914-1970) durante aproximadamente 20 (vinte) anos, entre 1938 a 1958, segundo Stanislav Kreichi (2012), compositor russo que se integrou ao laboratório de Murzin em 1961. Trata-se de um instrumento fotoelétrico/espectral e foi usado por vários compositores russos, entre eles Eduard Artemiev (1937), que o usou para fazer trilhas sonoras para uma série de filmes de Andrei Tarkovsky (1932-1986). Foi usado também no filme *Solaris* [Solaris] (1972) para criar uma sonoridade espacial. Atualmente, o desenvolvedor de *software* russo Alexander Zolotov fez o VirtualANS[3], uma emulação de *software* do ANS original, que "estende os princípios do original em muitas novas direções", segundo Warren Burt (2014)[4].

Outros instrumentos com novas interfaces surgiram durante a primeira metade do século XX, como o Telharmonium[5], construído por Thadeus Cahill (1867-1934) em 1897, o Mixtur-Trautonium inventado em 1929 por Friedrich Trautwein (1888-1956) em Berlim, o Ondas Martenot inventado em 1928 por Maurice Martenot (1898-1980) que foi utilizado em composições de Boulez, Messiaen, Varèse etc. e outras interfaces elertrônicas surgiram, mas não necessariamente para serem usadas em música.

Atualmente, para a criação de interfaces musicais, Miranda e Wanderley (2006) realizam uma pesquisa de desenvolvimento de novos dispositivos para a performance musical e apontam que os gestos de controle não possuem forma definida e podem ser os mesmos para instrumentos acústicos ou completamente novos para instrumento com formato alternativo, ou ainda, para instrumento expandido por adição de sensores chamados de *hyperinstruments* segundo Machover (1992). Em Miranda e Wanderley (2006, p. 103 a 164) há uma introdução aos sensores para adaptar em computadores. Apresentam mais de 20 (vinte) deles e apontam a sua aplicação no projeto de vários instrumentos musicais digitais. Também discutem métodos

[2] Inspirado neste compositor e suas sinestesias associativas de cores às notas musicais. http://en.wikipedia.org/wiki/ANS_synthesizer. Acessado em Abril de 2015.
[3] O VirtualANS é Gratuito em sistemas Linux, Windows e MAC OSX e pago em iOS e Android. http://www.warmplace.ru/soft/ans/. Acessado em Abril de 2015.
[4] http://soundbytesmag.net/virtual-ans-rebirth-of-a-classic-with-new-bells-and-whistles/. Acessado em Abril de 2015.
[5] http://en.wikipedia.org/wiki/Telharmonium. Acessado em Abril de 2015.

para converter os sinais em dados que podem ser usados para controlar o som em tempo real de síntese por *software*. Várias interfaces com sensores são apresentados e inclui modelos que usam o protocolo aberto de controle de som via rede denominado OSC (*open sound control*)[6] que pode substituir o protocolo MIDI (*Musical Instrument Digital Interface*)[7].

Na década de mil novecentos e oitenta, Richard Stallman[8], um dos criadores da Licença GPL (*general public license*)[9], criou um mecanismo legal para que todos pudessem desfrutar dos direitos de copiar, redistribuir e modificar *software*. Hoje, existe ampla gama de aplicativos livres voltados à prática de estúdio, sintetizadores, produção musical e programação em linguagem computacional de síntese sonora em tempo real que traz de modo oportuno a criação de novas interfaces musicais com um baixo custo. A licença do *Software* Livre em sua versão mais atual é a GPLv3 por Smith (2007). Dentre algumas linguagens livres de programação computacional voltada a síntese sonora que nasceram ao longo de anos de pesquisa e ainda são atualizadas nos dias de hoje encontramos *Csound*, *Puredata* e *superCollider* descritos abaixo.

Em 1957 na Bell Labs, Max Mathews desenvolveu o MUSIC, o primeiro programa para síntese sonora. Durante a segunda metade do século, Max foi líder na pesquisa de áudio digital, síntese e interação humano-computador. Na versão MUSIC V o programa obteve portabilidade ao ser compilado na linguagem FORTRAN IV e teve seu código aberto para pesquisadores. Depois, os algoritmos usados em MUSIC V serviram de modelo para um novo programa com base na linguagem "C" por Barry Vercoe em 1985[10], que o denominou *Csound,* distribuído também com código aberto. Funciona nos principais sistemas operacionais como MAC, MS-Windows e Unix/Linux. Atualmente esta linguagem computacional possui ferramentas para receber e enviar dados OSC e podemos encontrar seu manual em sua versão mais atual por Vercoe (2014).

Miller Puckette (2006), criador do *software* proprietário MAX/MSP entre 1993 e 1994 com David Zicarelli no IRCAM, em seu livro sobre o uso de técnicas eletrônicas para gravação, processos de síntese e análise de sons musicais, afirma que o termo "Música de computador", deveria atualmente ser chamado de "música eletrônica usando computador" e

[6] Um novo protocolo para a comunicação entre computadores, sintetizadores de som e outros dispositivos multimídia que é otimizado para a tecnologia de rede moderna, (Freed, 2014).
[7] http://www.midi.org/index.php. Acessado em Abril de 2015.
[8] https://stallman.org. Acessado em Abril de 2015.
[9] http://www.gnu.org/licenses/quick-guide-gplv3.pdf. Acessado em Abril de 2015.
[10] http://www.ime.usp.br/~kon/MAC5900/aulas/Aula6.html. Acessado em Abril de 2015.

diz ainda que a maioria das ferramentas de *computer music* disponíveis têm antecedentes em gerações de equipamentos anteriores (analógicos). Neste livro em especial, Puckette utiliza outro *software* de sua criação denominado *PureData*[11] (PD), que possui um sistema de conexões semelhante ao MAX/MSP, porém, é *Software* Livre e opera em qualquer sistema operacional em computadores, *smartphones* e *tablets* com sistema *android*. O PD também possui módulos externos para se conectar com *Arduino*[12], um dispositivo muito usado em projetos de robótica que é, basicamente, um micro-controlador programável capaz de se conectar com qualquer tipo de sensor ou dispositivo eletrônico, sendo então uma poderosa interface interativa (KREIDLER, 2009).

SuperCollider[13] é uma linguagem de programação para tempo real em síntese de áudio e composição algorítmica. O intérprete de sua linguagem é executado em multiplataforma (*OS X* / *Linux* / *Windows*) e se comunica via OSC com um ou mais servidores de síntese. O servidor de síntese *SuperCollider* é executado em um processo separado, ou mesmo em uma máquina separada por isso é ideal para música em rede em tempo real. O *SuperCollider* foi desenvolvido por James McCartney e originalmente lançado em 1996 sob os termos da GNU (*General Public License*) em 2002, quando McCartney se juntou à equipe da *Apple Core Audio*. Ele agora é mantido e desenvolvido por uma comunidade ativa e entusiasmada e é usado por músicos, cientistas e artistas que trabalham com som.

Em nossa atualidade, podemos encontrar em *smartphones* e *tablets* o conhecido sistema ANDROID que funciona alicerçado pelo núcleo LINUX[14] concebido por Linus Torvalds. O núcleo do Linux é um dos exemplos mais proeminentes de *Software* Livre, pois pode prover alicerce para o desenvolvimento e execução de outros *softwares* Livres. No sistema ANDROID existem aplicativos que fazem uso do protocolo OSC para enviar dados de sensores como: magnetômetro, giroscópio, acelerômetro, GPS (*Global Positioning System*), etc. até metáforas gráficas em tela sensível ao toque. Esses dados podem ser enviados pelos dispositivos de rede sem fio desses aparelhos que a cada dia são atualizados com novos sensores. Existem aplicativos comerciais que fazem uso do protocolo OSC que é aberto e estão disponíveis também nos dispositivos APPLE. Utilizar esses dispositivos

[11] http://puredata.info. Acessado em Abril de 2015.
[12] http://www.arduino.cc. Acessado em Abril de 2015.
[13] http://supercollider.github.io/. Acessado em Abril de 2015.
[14] http://pt.wikipedia.org/wiki/Linux_(núcleo). Acessado em Abril de 2015.

portáteis e seus sensores em substituição às interfaces a serem desenvolvidas com *Arduino* são uma opção viável para a pesquisa de novos controles para o gesto na execução musical.

A interação do homem com a máquina, fornece possibilidades incalculáveis de pesquisa sonora para o compositor que transporta a elaboração da partitura musical, para o algoritmo em linguagem computacional, e ainda, da improvisação com um instrumento, à expressividade com novos gestos para tocar uma interface. No estudo de algoritmos de interfaces desenvolvidas em *Software* Livre, que possibilitam o controle de síntese ou transformação do som em tempo real, encontramos trabalhos do compositor Iain McCurdy[15] que disponibiliza em seu site um grande número de algoritmos em *Csound* criados por ele, voltados à síntese em tempo real e os denomina "*Csound Realtime Examples*" (exemplos para tempo real em *Csound*). No estudo desse trabalho, encontramos *layouts* com metáforas na tela do computador e seu controle é através do *mouse*. Como os códigos são abertos, seguindo os princípios do *Software* Livre, existem possibilidades de troca dessas metáforas de imagem por gestos corporais usando sensores de movimento como acelerômetro, giroscópio, etc.

Diante desta contextualização nos perguntamos:

Por que a maioria das pesquisas sobre interação homem-máquina são realizadas com MAX/MSP e computadores APPLE?

Seria possível um compositor fazer uso de um computador com preço popular para a composição de música eletroacústica e implementação de novos gestos no controle da síntese sonora sem gastar com *Software* ou se envolver com pirataria?

Todos os recursos encontrados em *Software* Livre são suficientes para a composição eletroacústica na busca pela interação humano-computador e no desenvolvimento de novas formas de controle de síntese?

Em um levantamento sobre os trabalhos de pesquisa realizados neste campo encontramos motivos para esses questionamentos, pois, em trabalhos encontrados no *site* do IRCAM a maioria é realizado em MAX/MSP e mesmo o OM (*Open Music*)[16], desenvolvido pelo próprio IRCAM com código aberto, é executado em ambiente MAC OS em cursos oferecidos por Grégoire Lorieux e Carlos Agon. As teses e dissertações encontradas nas bibliotecas digitais da UNICAMP (Universidade de Campinas)[17] e da USP (Universidade de

[15] http://iainmccurdy.org/csound.html. Acessado em Abril de 2015.
[16] http://repmus.ircam.fr/openmusic/home. Acessado em Abril de 2015.
[17] http://www.bibliotecadigital.unicamp.br. Acessado em Abril de 2015.

São Paulo)[18] e da UFMG (Universidade Federal de Minas Gerais)[19] na área de composição eletroacústica possuem foco em alguns *Softwares* Livres como *Csound*, *Puredata*, *Open Music* e *SuperCollider*, porém, geralmente executados em sistemas operacionais pagos. Em outro exemplo, temos os trabalhos de César Traldi (2009) e Gabriel Rimoldi de Lima (2013) que desenvolveram programas em *Puredata* para suas composições, mas não usam um sistema operacional livre.

Com o objetivo de utilizar *Software* Livre desde sua plataforma, o Linux, realizamos composições de música eletroacústica que fazem uso de interfaces musicais interativas pré-existentes com adaptações para novos gestos ou desenvolvidas em linguagem *Csound* como composições algoritmicas. A abertura dos códigos das interfaces se faz necessária para o estudo dos intérpretes, suas possíveis adaptações de gesto e estudo para compositores. Portanto, a composição musical, a exposição e explicação desses códigos estão expostos neste trabalho como comprovação da viabilidade na utilização de programas abertos desde seu alicerce. Os algoritmos das interfaces/composições estão no ANEXO A.

Dentre essas composições criadas durante o período do curso de mestrado, 5 (cinco) composições musicais fazem uso de interfaces interativas, algumas delas foram criadas durante estudos da linguagem *Csound* e outras, com programações de outros desenvolvedores/compositores. Essas composições se intitulam:

1. *Vênus e Marte*, para *Theremin Giroscópico* e *Sintetizador por Randomização de Harmônicos*, ambas interfaces deste compositor.
2. *Caverna da Medusa*, para 6 (seis) interfaces, 2 (duas) por este compositor e 4 (quatro) interfaces de Iain McCurdy.
3. *Saigai to Higai* - 災害と被害, para acelerômetro e composição algoritmica.
4. ANS Estudo nº1 criado com VirtualANS em tela *multi-touch*.
5. ANS Estudo nº2 criado com VirtualANS em tela *multi-touch*.

Os próximos 5 (cinco) capítulos se enumeram e se nomeiam de acordo com a ordem das composições musicais citadas acima e suas nomenclaturas. As interfaces computacionais, que são os *layouts* das programações em *Csound* serão descritas dentro de seus respectivos subcapítulos. As partituras estão no ANEXO B e uma lista dos *Softwares* Livres usados na montagem das partes acusmáticas, partituras, etc. se encontra no ANEXO C.

[18] http://www.teses.usp.br. Acessado em Abril de 2015.
[19] http://www.bibliotecadigital.ufmg.br. Acessado em Abril de 2015.

1. VÊNUS E MARTE

A motivação poética da composição de *Vênus e Marte* começou no desenvolvimento de interfaces interativas. Estas interfaces são o resultado de composições algorítmicas em linguagem computacional (*Csound*) que o compositor depois de desenvolvê-las, precisa aprender a tocá-las para guardar seus novos "objetos sonoros" (Schaeffer, 1994), que são os princípios de construção adotados neste trabalho.

A interface *Sintetizador por Randomização de Harmônicos* que virá detalhada mais abaixo, é o instrumento harmônico da música, o elemento que relacionamos poeticamente ao masculino, gerador de milhões de sementes, Marte o "Deus da Guerra" e também Marte o planeta. Maiores motivos pela escolha do título da obra se encontram dentro da música quando nasce Harmonia[20].

O *Theremin Giroscópico* foi associado ao lado feminino, dentro dessa intenção poética, e à qual relacionamos a Vênus. O timbre é similar ao Theremin original, porém com efeito de eco "ping pong" e reverberação, muitas vezes lembra voz de mulher e mesmo em regiões mais graves, ela é o cântico melódico acompanhado com harmônicos cordais[21], que nos remete aos trovadores que estão presentes em todos os períodos da história da música. A semelhança com a conjuntura a ser evitada se faz pela tentativa de criar o novo pela transfiguração do velho, uma proposta de conceito que nomeamos "Estética da Reciclagem".

Podemos considerar o princípio básico de *Vênus e Marte* como sendo duas canções com uma peculiar textura de melodia acompanhada que realmente é uma composição à duas vozes, pois, Marte acompanha Vênus com notas sucessivas mergulhadas num mar de reverberação. A parte do *Sintetizador por Randomização de Harmônicos* foi gravada e montada com o *software* Qtractor (ANEXO C). A partitura para performance ao vivo contém a imagem espectrográfica dessa parte fixa obtida do *Software* Spek (ANEXO C) para o auxílio visual na leitura das notas a serem tocadas pelo performer do *Theremin Giroscópico*.

Composicionalmente, a música trata de improvisos com as duas interfaces utilizadas, sendo o processo composicional interativo no qual evita a sonoridade estocástica de equações geradoras de sons e privilegia o discurso direto pelo ato de expressão do compositor. Não se tratam de improvisos absolutamente aleatórios, mas sim, recolhimentos e

[20] Harmonia, filha mortal de Vênus e Marte de acordo com mitologia romana.
[21] A série harmônica que se relaciona diretamente aos números inteiros.

montagem posterior no caso das sonoridades do *Sintetizador por Randomização de Harmônicos*. A parte do *Theremin Giroscópico* foi cuidadosamente criada durante audições da parte fixa criada anteriormente e utilizando o próprio instrumento/dispositivo para ouvir e definir o seu discurso. De fato, a técnica se assemelha à criação de música popular em estúdio, porém, aplicada à construção de uma narrativa não mimética, objetivando a estética da música eletroacústica.

A utilização de *software* Livre para esta composição foi amplamente satisfatória e se resume aos seguintes programas: *Csound* para a síntese em tempo real e conexão via OSC para os controles. Servidor de áudio *Jack Control* para a gravação e edição em *Qtractor*. *Spek* para a impressão do espectro da parte fixa e *Gimp* para a edição das imagens do espectro para a confecção da partitura. Detalhes destes programas se encontram no ANEXO C.

O áudio da parte fixa para as futuras performances e outros arquivos relativos à obra *Vênus e Marte* estão disponíveis abaixo por *url* e seu respectivo *QrCode* para facilitar a leitura e visualização por dispositivos portáteis. A gravação em audiovisual da estréia no SEMPEM XIII também possui *url* e QrCode abaixo. Todos esses arquivos também se encontram na mídia CD-ROM anexo ao final deste trabalho.

Arquivos relativos à música:

https://drive.google.com/folderview?id=0Bx_5iQ2UHJJvWDN0SkhrT1hzUmM&usp=sharing

Gravação em vídeo no SEMPEM XIII:

http://youtu.be/-cKeY7We_Mg

1.1 Sintetizador por Randomização de Harmônicos

Durante estudos em tutoriais de *Csound*, nos deparamos com possibilidades gráficas, tanto para visualização de osciloscópio, lissajous, poincaré, etc. como de gráficos para interação, no sentido que descreve Almeida (2011), trazendo as metáforas do knob, botão, *slider*, fios de conexão... tudo que representemos virtualmente no computador.

Com o *slider* (potenciômetro deslizante) e o *mouse* como ferramenta de interação, podemos alterar um dado de uma equação, por exemplo. Esse controle que faz uso de mouse é limitado e se iguala à interação com apenas um dedo na tela. As telas de toque múltiplo ampliam para além de dez (10) dedos diretamente na tela (*multi-touch screen*). Em termos musicais, é a transformação de um instrumento melódico em instrumento harmônico.

Outro exemplo de *slider* que podemos usar como controle é o giroscópio eletrônico que nos fornece três (3) dados variáveis, os ângulos das rotações nos eixos x, y e z. Enfim, qualquer sensor que emita uma variação de dados, pode ser usado como controle na interação entre homem e máquina para a manipulação da síntese sonora em tempo real.

Da necessidade de ouvir um som que demonstre que harmônicos senoidais tocados de forma sucessiva podem fazer soar sua vibração fundamental pela força da reverberação, foi criada uma interface com cinco (5) *sliders*, demonstrada na figura 01 e que funciona da seguinte forma:

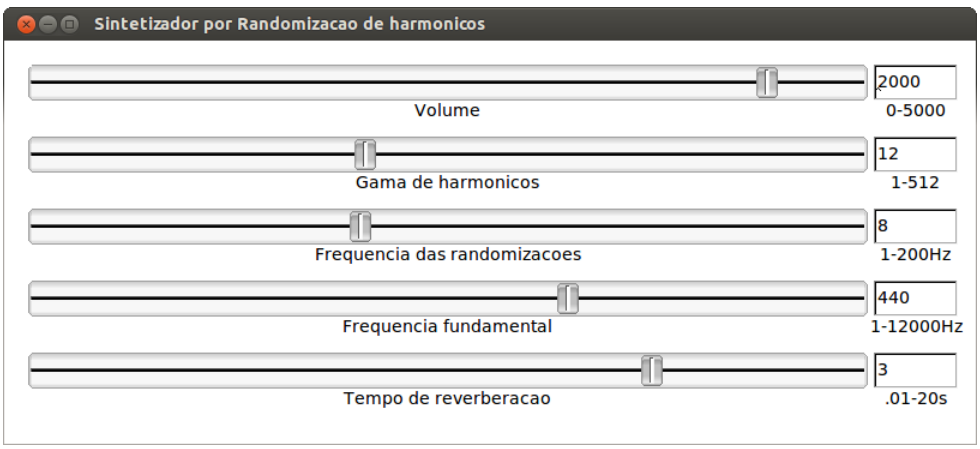


Figura 01: Captura de tela do *Sintetizador por Randomização de Harmônicos* em *Csound*.

1. O primeiro *slider* foi associado diretamente à amplitude do único oscilador deste algoritmo (*opcode* "**oscili**" do *Csound*), que suporta valores de 0 a 32768 por *default*,

caracterizando-se como volume geral. Limitamos o máximo em 5.000 (cinco mil) para evitar que o total ultrapasse o limite de 32.768 por efeito da reverberação.

2. O segundo *slider* (Gama de harmônicos) controla o valor máximo para o sorteio do randomizador, melhor dizendo, após buscar um comando de *Csound* que utilize sorteio pseudo-aleatório, encontramos diferentes comandos que utilizam *random*, o melhor deles para este propósito foi o "**randomh**" que suporta 3 parâmetros: um valor mínimo, um valor máximo e um valor para a frequência dos sorteios. O valor mínimo ficou fixo em um (1) , pois este é o primeiro harmônico, o valor máximo foi limitado em 512, portanto este deslizante (*slider*) nos possibilita variar entre 1 e 512. Isso é um exagero em números de harmônicos cordais, mas pode ser alterado dentro do algoritmo facilmente.
3. O terceiro *slider* (Frequência das randomizações) cuida do terceiro parâmetro do "**randomh**", determinando a velocidade dos sorteios em ciclos por segundo. Como vemos na figura 01, o valor máximo de 200Hz se faz presente em respeito às regras de síntese granular (LEE, 2000), onde o menor grão recomendável, 0,005s (cinco milésimos de segundo) vem da fração 1/200.
4. O quarto *slider* (Frequência fundamental) foi associado à frequência de nosso único oscilador, o segundo parâmetro do comando "**oscili**", poderíamos utilizar um teclado MIDI para fornecer essas frequências, porém estaríamos limitados a vários mimetismos: do ato de tocar teclas, de doze (12) notas por oitava, 27,5Hz (vinte e sete Hertz e meio) como frequência mais grave, etc. Como vemos na figura 01 a frequência mais grave possível é 1Hz (um hertz) e a mais aguda 12KHz (doze mil Hertz). Estes valores são multiplicados diretamente pelo que nos retorna o "**randomh**" fazendo com que o parâmetro para frequências no comando "**oscili**" permaneça em constante alteração conforme frequência determinada no terceiro *slider*.
5. O quinto *slider* controla o T_{60} (tempo de atenuação em 60dB) da reverberação acoplada à saída do oscilador, como se houvesse uma hipotética sala com o poder de mudar o coeficiente de absorção de suas paredes. A reverberação é responsável pelas intersecções entre as notas sucessivas da granulação de harmônicos causada pela constante alternação de números inteiros que multiplicam a frequência fundamental do quarto *slider*, isso faz a soma desses parciais harmônicos no tempo, o que nos dá a sensação de ouvir a frequência fundamental.

Devido aos números advindos do "**randomh**" incluírem frações, eliminamos os números após a vírgula com a função "**int( )",** restando apenas os números inteiros. Não conseguimos, porém, retirar os números após a vírgula na janela de monitoramento do segundo *slider*, causando estranheza quando vemos durante o funcionamento, mas não corresponde ao resultado sonoro.

Sem os números inteiros o sintetizador produz massas inarmônicas, o que também é interessante e neste caso seria melhor incluir um sexto *slider* para o mínimo valor em "**randomh**", mas isso já é instrumento para outra composição ou outro instrumento dentro da composição.

Da forma como está disposto na figura 01, esta interface interativa foi alvo de constante estudo de suas possibilidades sonoras tal como um instrumento musical. Incluindo a calibragem de alguns valores inclusos nele durante o ato de criar e interagir, como o ato de afinar um instrumento. Dentro destas experimentações, acumulou-se com o tempo, vários objetos sonoros que integraram a composição da parte fixa de *Vênus e Marte*.

A programação em *Csound,* que é estritamente textual, possui editores para seu texto (Qutecsound[22], Winxound[23], etc.) que reconhecem sua linguagem de programação e subdividem por cores as palavras-chave do código para o auxílio visual durante esta edição textual. Ao executarmos o código, aparece na tela, a interface da figura 01 como descrevemos acima. O código desta interface se localiza no ANEXO A em algoritmo nº 1 e em sua forma executável pelo arquivo disponibilizado em CD-ROM e *on-line* pelo *link* dos arquivos relativos à música, página 28.

Para fins de estudo de *Puredata*, foi criada uma versão do *Sintetizador por Randomização de Harmônicos* que apresentamos na figura 02 e descrevemos logo abaixo algumas comparações desta linguagem com a linguagem *Csound.*

[22] http://qutecsound.sourceforge.net. Acessado em Abril de 2015.
[23] http://winxound.codeplex.com. Acessado em Abril de 2015.

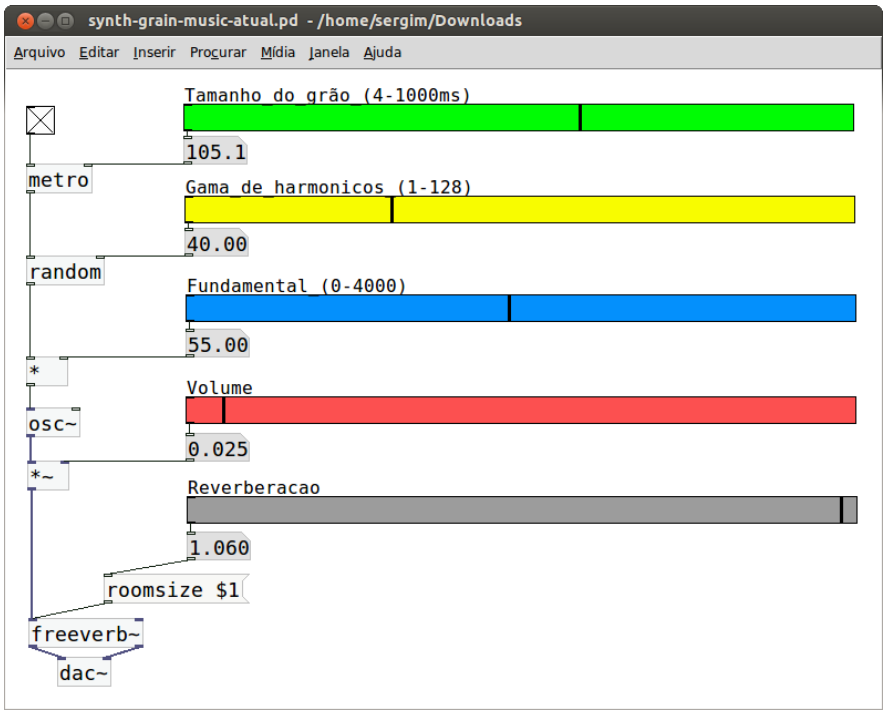


Figura 02: Captura de tela do *Sintetizador por Randomização de Harmônicos* em *Puredata*.

Na figura 02, temos a programação por fluxograma do *Puredata*. Uma vez montado o circuito, já estará funcionando, pois, nesta linguagem não temos as possibilidades de erro de sintaxe textual que impedem a programação funcionar, uma vantagem ao método textual e com um reduzido número de erros nas tentativas experimentais comparados ao *Csound*. A implementação de novos recursos ao circuito também se mostra bem simples em relação ao implemento em texto. O entusiasmo gerado por essas facilidades faz com que tenha um grande número de desenvolvedores criando bibliotecas adicionais de objetos como o "**freeverb~**" do exemplo acima.

1.2 Theremin Giroscópico

Na página pessoal de Iain McCurdy citada na introdução, encontramos um exemplo que inclui interface visual em FLTK (*Fast Light Toolkit*)[24] denominado Theremin, criado em 2006 conforme consta em seu algoritmo.

Esta interface simula o som do instrumento original Theremin, não inclui o modo peculiar de tocá-lo, mas, oferece a seguinte solução:

[24] http://www.fltk.org/index.php. Acessado em Abril de 2015.

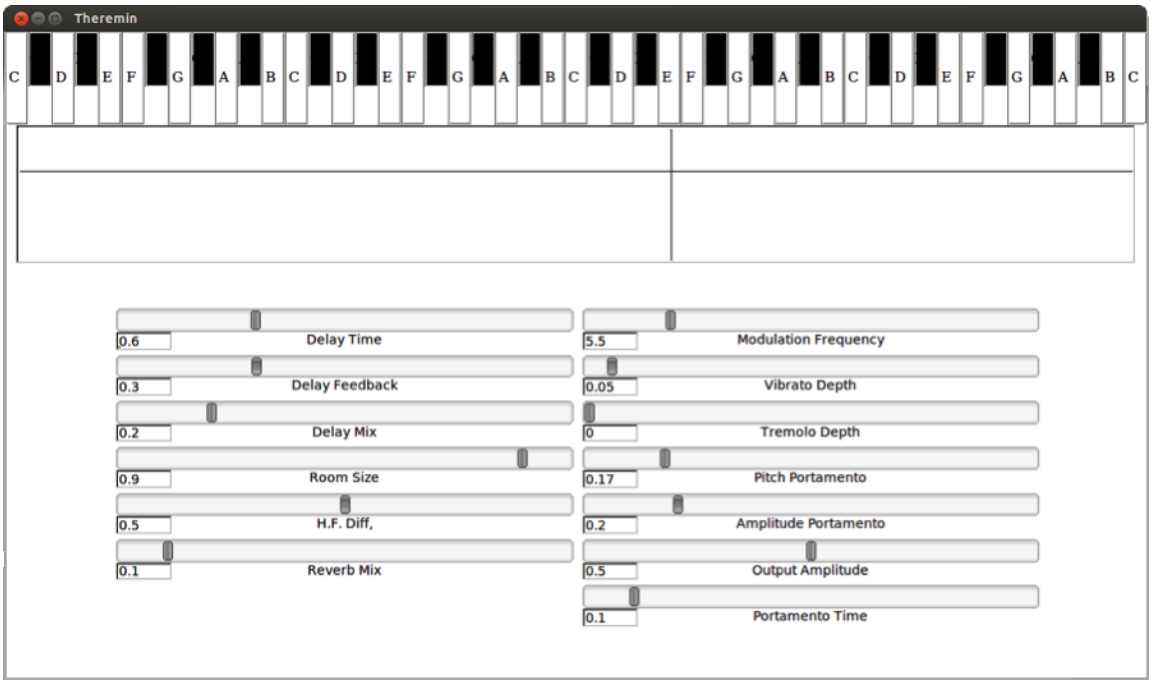


Figura 03: Captura de tela do Theremin de Iain McCurdy em *Csound realtime examples* do diretório: Miscellaneous.

Com o mouse é possível "manipular" as metáforas dos potenciômetros de "Delay time", "Delay Feedback", etc. como vemos na figura 03, mas em especial, a forma de tocar o instrumento é por meio da janela que contém um vetor bidimensional (*joystick*) logo abaixo das teclas que servem como referência visual. Ao clicar nesta área o instrumento soa e surgem as linhas vertical e horizontal determinando exatamente o ponto clicado. A linha vertical nos indica a posição no eixo "X" associado à frequência das notas. A linha horizontal indica a posição no eixo "Y" associado à intensidade.

Conhecida esta interface, pensamos em um modo de controlar esses vetores de frequência e amplitude de outra forma que não fosse pelo mouse do computador. Com sensores de proximidade, por exemplo, poderíamos imitar a forma original de "tocar" o Theremin. Apesar que, com o mouse, seja mais fácil do que o próprio Theremin.

Por meio do protocolo aberto de controle de som via rede de computadores o OSC, referido na introdução, é possível enviar dados variantes como os citados acima, para qualquer computador que esteja em uma mesma rede. Este protocolo já está sendo amplamente usado em substituição ao protocolo MIDI e oferece pelo menos 2 (duas) inovações em relação a este antigo sistema (década de 1980). Primeiro, as variáveis não são de apenas 7 (sete) bits (com valores de 0 a 127) e segundo, funcionam com rede sem fio.

Em dispositivos celulares e *tablets* é possível encontrar aplicativos especialmente desenvolvidos sob o protocolo OSC fazendo com que possamos enviar os dados dos sensores destes aparelhos para um computador sem usar fio.

Em especial, o *Theremin Giroscópico* utiliza um aplicativo denominado *AndOSC*, disponibilizado de forma gratuita para sistemas ANDROID. É preciso instalar este aplicativo em um dispositivo que contenha giroscópio de 3 (três) eixos e bússola de magnetômetro, geralmente encontrados em aparelhos com GPS (*Global Positioning System*).

A interface do aplicativo é simples, possui na tela a visualização dos seguintes dados variantes como vemos na figura 04a.

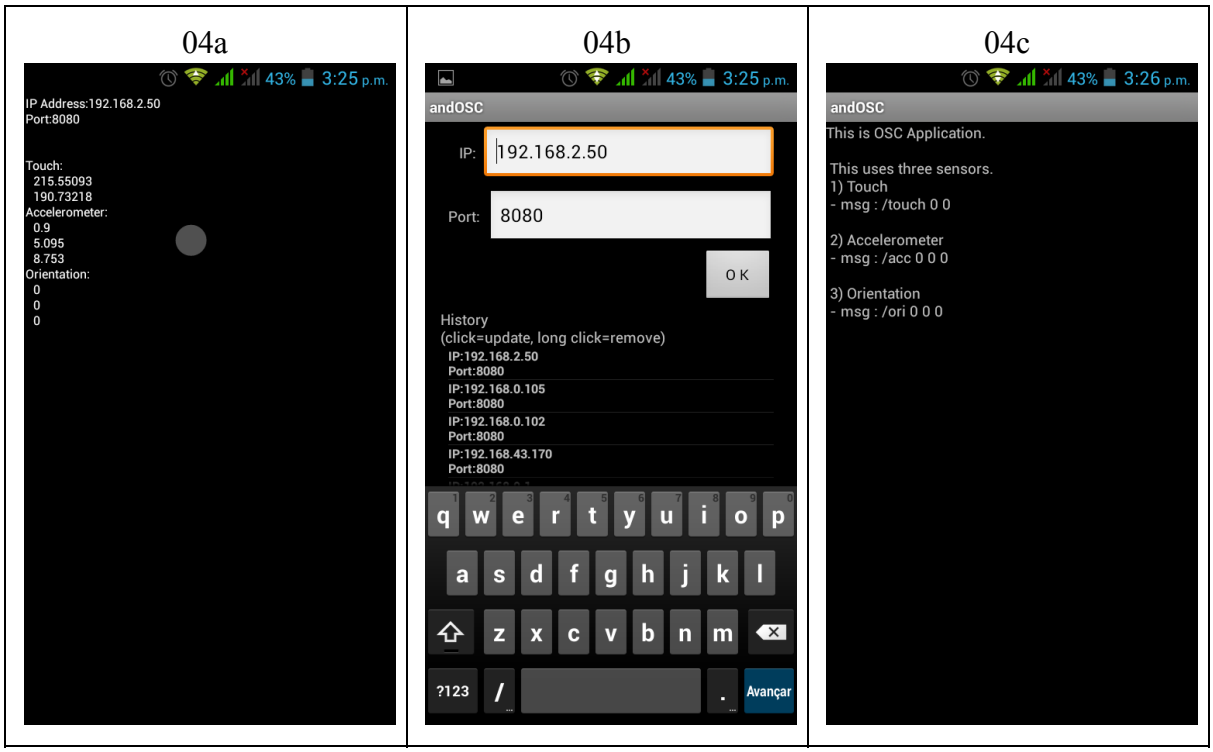


Figura 04: 04a - Captura de tela do aplicativo *AndOSC* enviando dados; 04b - Tela de Configurações; 04c - *Help* do aplicativo.

1- Do touchscreen, a posição na tela (círculo cinza) pelos eixos "X" e "Y".

2- Do acelerômetro, a movimentação do aparelho pelos eixos "X", "Y" e "Z".

3- Do giroscópio (orientation), a rotação do aparelho pelos eixos "X", "Y" e "Z".

Todos esses dados são enviados para o endereço TCP-IP (*Internet Protocol*) e uma "porta", que pode ser determinado na opção de configurações demonstrada na figura 04b.

Na figura 04c o aplicativo informa ainda, num breve *help*, as nomenclaturas a serem usadas no algoritmo de *Csound* ou *patch* de *Puredata*: /touch f f, /acc f f f e /ori f f f que configuram o protocolo OSC e dão nomes aos dados enviados pelos sensores.

Com isto, substituimos os vetores originais do instrumento Theremin em *Csound* pelos dados do sensor giroscópico da seguinte forma:

Estudando o algoritmo de McCurdy disponível em sua página, encontramos as variáveis responsáveis pelos eixos "X" e "Y" da programação, que são: "**gkoctx**" e "**gky**" encontradas na linha 28 na declaração do *opcode* "**FLjoy**":

```
gkoctx, gky, ihandlex, ihandley    FLjoy " ", 7, 11, 0, 1024, 0, 0, -1, -1, 1200,150, 13,100
```

Esta linha não precisa ser alterada, mas é preciso mudar os dados das variáveis "**gkoctx**" e "**gky**" antes de serem usadas. Isso só é possível porque a linguagem *Csound* contém *opcodes* voltados para OSC que são: "**OSCinit**" e "**OSClisten**" que foram implementados no algoritmo de Iain McCurdy como segue abaixo:

Acrescentado ao cabeçalho, junto às definições de taxa de amostragem:

```
giosc1 OSCinit 8080
```

Acrescentado dentro do instrumento 1 (um) logo após sua abertura:

```
kX init 0
kY init 0
kZ init 0
kans1 OSClisten giosc1, "/ori", "fff", kX,kY,kZ
gkoctx = 7+(kX/90)
gky = kZ*12
```

Para esclarecer o código acima definimos a abertura da porta 8080 (este número precisa coincidir com o número da porta configurada no aplicativo *AndOSC*) na inicialização do comando "**OSCinit**" e em seguida, declaramos dentro do instrumento, as variáveis de controle kX, kY e kZ, depois associamos, por meio do comando "**OSClisten**", às três

variáveis “fff” que vêm através da rede com o nome de “/ori” de acordo com a informação do *help* do *AndOSC*.

Depois, aplicamos uma equação simples no “**gkoctx**” e “**gky**” para adaptar à variação dos sensores em kX e kZ respectivamente. Uma das soluções citadas por Miranda e Wanderley onde equações podem fazer o encaixe dos dados de saída em dados de entrada.

A variável “kX” assume então a variação de valores entre 0º e 359º, que são os 360º (trezentos e sessenta graus) do ângulo de Azimute. Esses valores serão responsáveis pela variação de frequência do instrumento que tem o âmbito de 4 (quatro) oitavas, portanto uma oitava a cada 90º (noventa graus). Com o instrumento (*tablet* ou *smartphone*) voltado para o norte (0º), soa o Dó 1. À medida que giramos no sentido horário, a frequência aumenta, Dó 2 em 90º, Dó 3 em 180º e Dó 4 em 270º como podemos ver na figura 05. Entre 359º e 0º temos um rápido glissando entre o Dó 5 e o Dó 1.

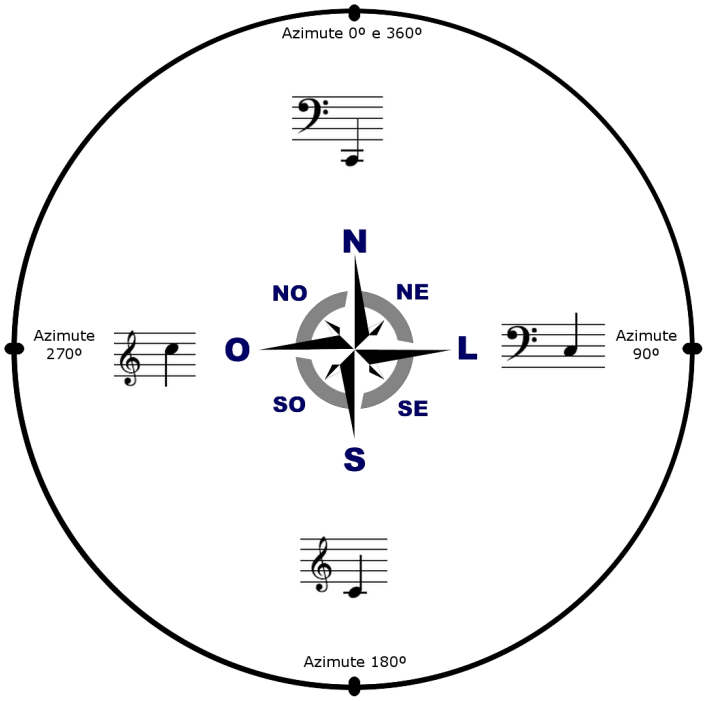


Figura 05: Relação de ângulos de Azimute para as notas do *Theremin Giroscópico*.

A amplitude do Theremin ficou definida pela variável “kZ” que foi a mais conveniente para o *tablet* usado na performance da gravação da obra *Vênus e Marte* na programação artística do SEMPEM XIII, porém pode ser trocada pela variavel “kY” de acordo com a conveniência de cada aparelho. O Código da programação que inclui o implemento citado acima se localiza no ANEXO A em algoritmo nº 2, *on-line* pelo *link* dos arquivos relativos à música, página 28 e no CD-ROM anexo.

2. CAVERNA DA MEDUSA

Caverna da Medusa foi composta à partir de 2 (duas) interfaces criadas durante estudos de *Csound* e 4 (quatro) interfaces criadas por Iain McCurdy que estão disponíveis em seu site. Se divide em duas partes, ambas criadas pelo método conhecido como "*playback*", onde os instrumentos são inseridos um a cada vez na composição, sendo possível ouvir cumulativamente as gravações anteriores.

No primeiro movimento foi gravado inicialmente o improviso com a interface "FoF" (função de onda formante), depois adicionado o improviso com a interface "FoFx6" sendo possível ouvir simultâneamente o improviso anterior e assim sucessivamente na seguinte ordem: "FoF", "FoFx6", "Theremin", "Harmonizador em Tempo Real", "XY/FM" e "Delay Doppler".

Durante a composição, a sonoridade inspirou o nome. Ela evocava lembranças de histórias sobre mitologia grega, porque os efeitos de vozes da interface "FoFx6" quando têm sua frequência reduzida até se tornar um pulso, parece congelar a voz no tempo e remetia às estátuas das pessoas transformadas em pedra depois de terem olhado nos olhos da Medusa. O som do "Delay Doppler", que lembra vento encanado, foi adicionado depois da idéia do nome, para trazer a sensação do mundo subterrâneo onde se passa o mito.

A segunda parte foi criada um tempo depois e a sequência das gravações das interfaces foi outra: "Delay Doppler", "FoF", "FoFx6" e depois o Theremin, "XY/FM" e "FoF" foram adicionados de forma pontual por toda a música aproveitando a possibilidade de edição não linear do *software* multi-pista *Qtractor*. Nessa parte não temos a presença do *Harmonizador em Tempo Real*, no intuito que na sonoridade total, sentíssemos falta de alguma coisa e relacionássemos à ausência da Medusa.

Mais uma vez, a criação desta obra se fundamentou em processos composicionais interativos tal como *Vênus e Marte*. A busca pelo retorno à expressividade do autor em contraponto à estruturação puramente matemática. A técnica de *playback* também utilizada na música popular serviu de base para a produção sonora, porém, com foco em sons inauditos. Muitos improvisos foram descartados até que fosse obtido um resultado satisfatório e também a edição não linear serviu de método estruturador dos objetos sonoros obtidos.

A utilização de *software* Livre se limitou às seis interfaces construídas com a linguagem *Csound* e o processo de gravação e edição se deram com o servidor de áudio de *Jack Control* aliado ao programa multi-pista *Qtractor* detalhados no ANEXO C.

A seguir, as descrições dessas 6 (seis) interfaces utilizadas na composição de *Caverna da Medusa,* serão na ordem das gravações da primeira parte. As 4 (quatro) interfaces do compositor Iain McCurdy, quando abertas, acompanham texto explicativo em inglês e foram traduzidas para o português em tradução livre para que não faltasse neste trabalho a descrição completa de todas as interfaces usadas ou criadas. Portanto, nos dispusemos a traduzir as interfaces de McCurdy. Os arquivos em áudio *mp3* da primeira e segunda parte e outros arquivos digitais da criação da música estão disponibilizados na *url* e seu respectivo *QrCode* abaixo, assim como em CD-ROM anexo.

https://drive.google.com/folderview?id=0Bx_5iQ2UHJJvRHRDSm82VmNPVXc&usp=sharing

2.1 FOF (Iain McCurdy)

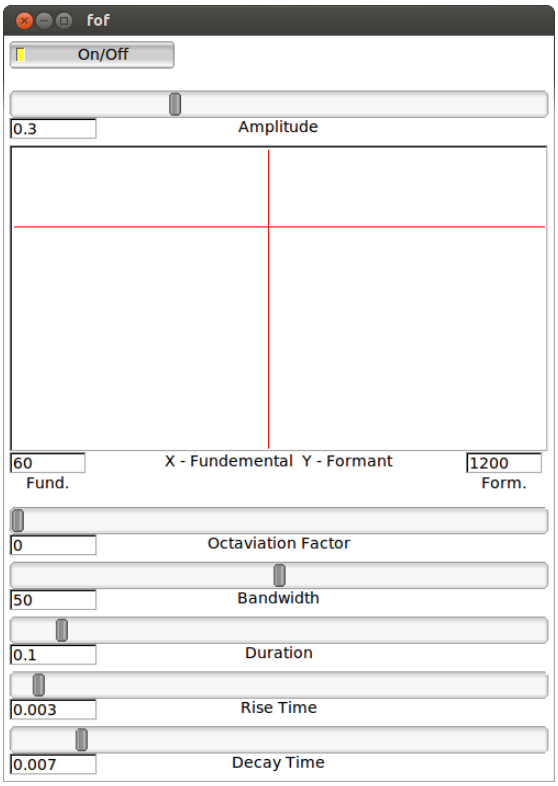


Figura 06: Captura de tela da interface FOF de Iain McCurdy em *Csound realtime examples* do diretório: Síntese Granular.

Fonction d'onde formantique (Função de onda formante - *FOF*) é um tipo bastante específico de síntese granular em que sua intenção é a criação de vogal vocal, que soe através do uso de grãos de onda senoidal repetidas rapidamente. Sua utilização não convencional não deveria ser vista como uma restrição. Se o usuário inicia o exemplo, a primeira coisa que se ouve é um fluxo de impulsos repetidos, em que cada pulso é facilmente discernível. O pulso de cada grão é perceptível isoladamente. Se o controle deslizante 'Formant' é movido, ouvimos que a afinação de cada grão é modulada. Se o controle deslizante “fundamental” é movido, lentamente da esquerda para a direita, ouvimos que a frequência de repetição de grãos aumenta. À medida que passamos cerca de 35 hertz, não somos mais capazes de distinguir grãos individuais e surge a percepção de tom, que é uma consequência dos grãos idênticos repetidos periodicamente. Mantendo “fundamental” em um valor alto (limite 200Hz), ao mover o controle deslizante “Formant”, o efeito é de um filtro de banda a ser aplicado ao som. Um formante é realmente apenas um pico de energia em um espectro de som harmônico. Os fenômenos demonstrados são os princípios por trás da síntese FOF. Para

sintetizar convincentemente, sons de vogais da voz humana, cerca de seis sinais *FOF* simultâneos são necessários. O exemplo *FOF x 6* demonstra isso. O envelope de amplitude, que é aplicado a cada grão, é controlado por uma combinação de "durações" (kdur), "Tempo de ascensão" (Kris), "tempo de decaimento" (kdec) e "largura de banda" (kband). A largura de banda é controlada por uma curva exponencial definida em uma função separada e aplicada ao decaimento de cada grão. "índice de Oitavação" (koct) é normalmente zero, mas como ele tende a anular qualquer outro grão, fica cada vez mais atenuado. Quando é exatamente uma densidade de grãos, é efetivamente reduzido pela metade e a fundemental "fof" é descartada por uma oitava. De 1 a 2, o processo é repetido e a densidade é de novo reduzida para metade e assim por diante do 2 a 3 e mais além. Este efeito é percebido de forma bastante diferente para texturas densas e esparsas.

2.2 **FOF x 6** (Iain McCurdy)

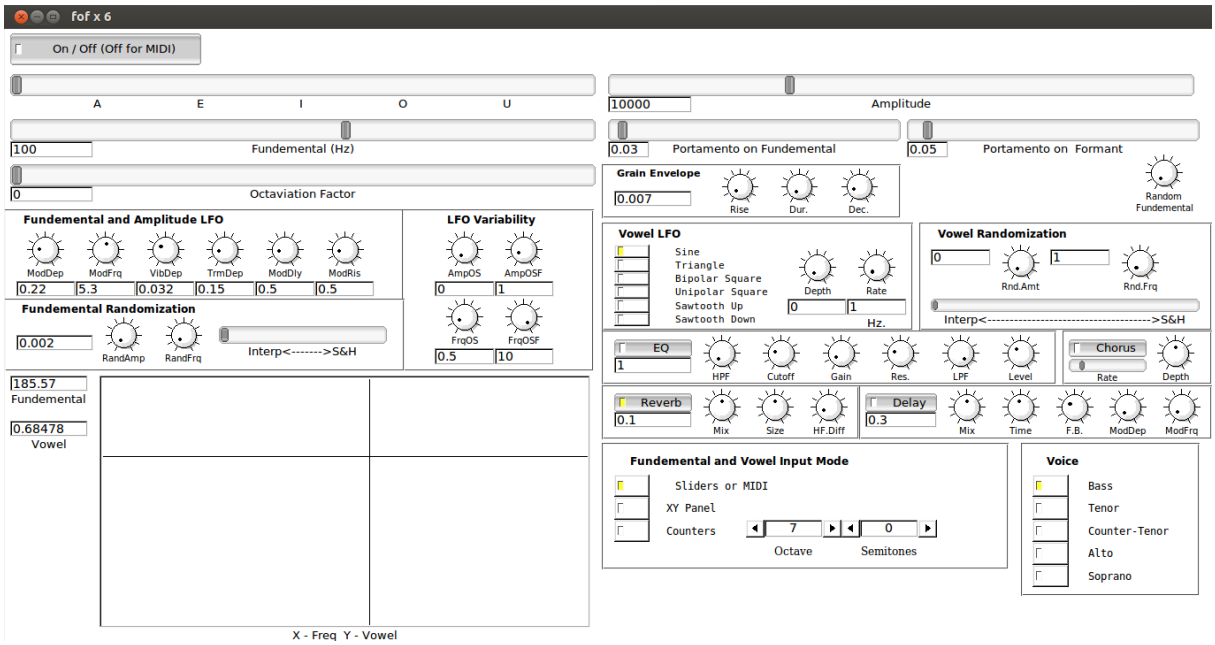


Figura 07: Captura de tela da interface FOF x 6 de Iain McCurdy em *Csound realtime examples* do diretório: Síntese Granular.

Esta interface faz uso de seis instâncias simultâneas do *opcode* para imitar uma variedade de vozes cantantes, cantando vários sons de vogais. A interface permite uma variedade de métodos para introduzir os valores dos parâmetros essenciais e o "*patch*" incorpora vários efeitos e elaborações adicionais. O primeiro método para a entrada de

parâmetro é “On/Off. Of for MIDI”. Para usar sliders, primeiro ative o botão “on/off”, que permite o controle da fundemental do tom usando o deslisante “Fundamental (Hz)” e da vogal usando o controle deslizante “A E I O U”. Para usar MIDI basta deixar o botão “on/off” desativado e tocar as notas no instrumento MIDI. O instrumento é monofônico com portamento entre as notas controláveis pelo deslisante “Portamento on fundamental”. O controlador MIDI 2 pode ser utilizado para modular o tipo de vogal produzido e um outro pode controlar o vibrato de profundidade. O segundo método para a introdução de parâmetros é pelo “Painel XY”. Com este método activado, uma fundamental vogal e de tom podem ser controláveis através do painel de XY na parte inferior da interface. Movimentando no plano vertical, varia a fundamental. Movimentando no plano horizontal, varia o tipo de vogal. Selecionando a opção “Contadores” para o modo de entrada de parâmetro, permite que a fundamental seja definida à partir dos dois contadores ao lado. Neste modo, o tipo de vogal é variada usando o controle deslizante. No painel frontal contém vários controles de modulação LFO fundamental e amplitude. “vibrato” refere-se a modulação fundamental e “tremolo” refere-se a modulação de amplitude. Existem controles para a rapidez com que a modulação constrói, (“ModRis”) e o tempo de atraso antes de começar a construir, (“ModDly”). Existe um controle geral de profundidade de modulação e freqüência (“ModDep” e “ModFrq”) e controle independente sobre vibrato e profundidade de tremolo (“VibDep” e “TrmDep”).

2.3 Theremin (Iain McCurdy)

A interface “Theremin” de Iain McCurdy não inclui texto explicativo do autor e não possui mais publicação em seu site, o algoritmo usado para nossa implementação de OSC data de 2006 e foi baixado da INTERNET antes de ter sido retirado do site. O algoritmo se localizava no diretório “*Miscellaneous*”, onde encontramos exemplos interessantes, porém, sem classificação. Melhores detalhes desta interface foi demonstrada acima na descrição técnica do *Theremin Giroscópico*.

Atualmente é possível encontrar uma versão aprimorada por desenvolvedores de um dos programas editores de *Csound,* denominado *QuteCsound* (ANEXO C). Na figura 08 abaixo, podemos verificar a diferença entre interfaces gráficas como os *widgets* de *QuteCsound* e a antiga interface de McCurdy que utiliza FLTK (*Fast Light Toolkit*) exposta na figura 03 no capítulo 1.

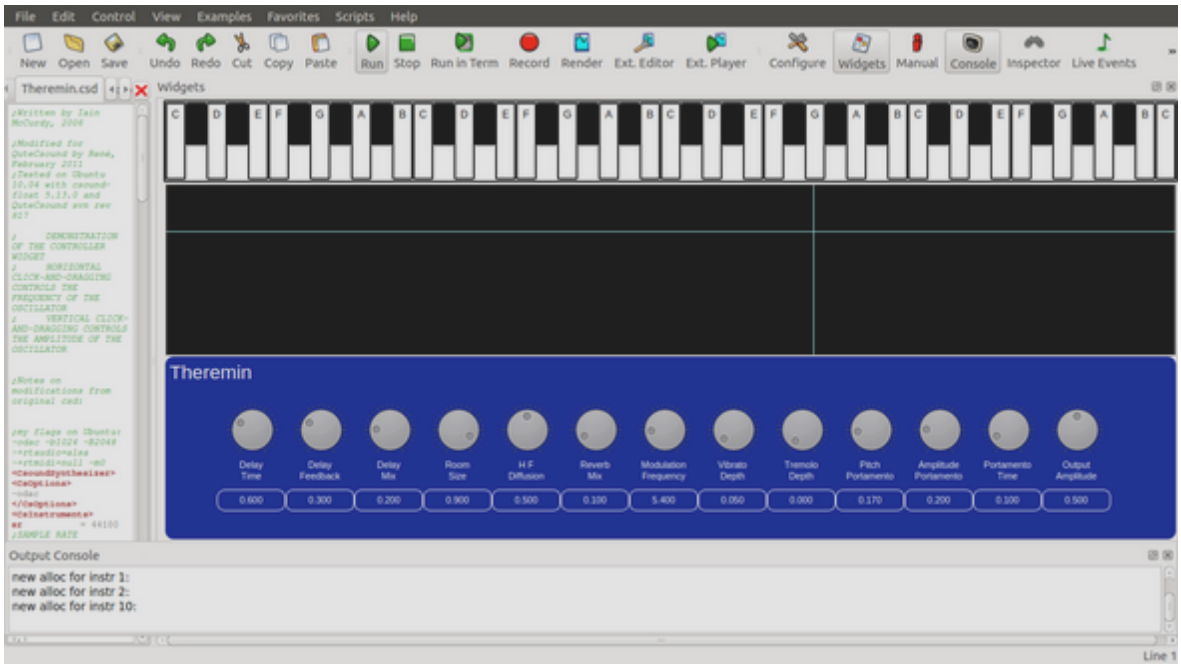


Figura 08: Captura de tela da interface Theremin de Iain McCurdy em *Csound realtime examples* do diretório: *Miscellaneous*. Adaptado por René em *QuteCsound*.

2.4 Harmonizador em Tempo Real

Baseado em apenas um *opcode* de *Csound* denominado "***harmon***", o *Harmonizador em Tempo Real* é uma sugestão do próprio Berry Vercoe no *Csound* manual, ao sugerir o uso de teclado MIDI para determinar as frequências de saída nos auto-falantes, independentemente das frequências de entrada, tornando-se um instrumento de teclas onde seu timbre é o som que entra no microfone em tempo real de execução.

Um detalhe importante desta interface na composição de *Caverna da Medusa* é a configuração do *imode*, modo de interpretação para as entradas de frequência, onde para 1: os valores de entrada são as frequências solicitadas reais em Hertz e para 0: os valores de entrada são proporções no que diz respeito à frequência do sinal de áudio analisados do sinal de entrada.

Na composição foi utilizada a configuração "0" onde as vozes extras criadas pela determinação nas teclas do teclado acompanham por paralelismo a altura das notas cantadas no microfone. Este método oferece menos distorção da voz porque evita grandes diferenças entre frequência de entrada e frequência de saída exigida. Na composição do primeiro movimento, improvisamos tétrades em torno do Dó central que está configurado para emitir

exatamente a frequência da voz que entra. Portanto a progressão harmônica desses acordes acompanham os glissandos da voz.

Possui ainda, várias tabelas de possíveis afinações, micro-afinações e macro-afinações que ainda são objetos de estudo deste autor. Na gravação da primeira parte de *Caverna da Medusa* foi utilizada a afinação justa, descrita dentro do algoritmo em sua função número 102 (cento e dois), sendo uma proposta de afinação justa para a escala cromática, pois em livros sobre o assunto encontramos dados apenas para a escala diatônica. Na figura 09 podemos ver o fluxograma desta programação que não possui interface visível em tela. Suas interfaces para a performance se reduzem ao microfone na mão direita do intérprete para a entrada de voz e a mão esquerda realiza acordes em torno da altura atual de sua voz com um teclado MIDI. Seu código está disponível no ANEXO A, em algoritmo nº 3, *on-line* pela *url* disponibilizada na página 38 e CD-ROM anexo.

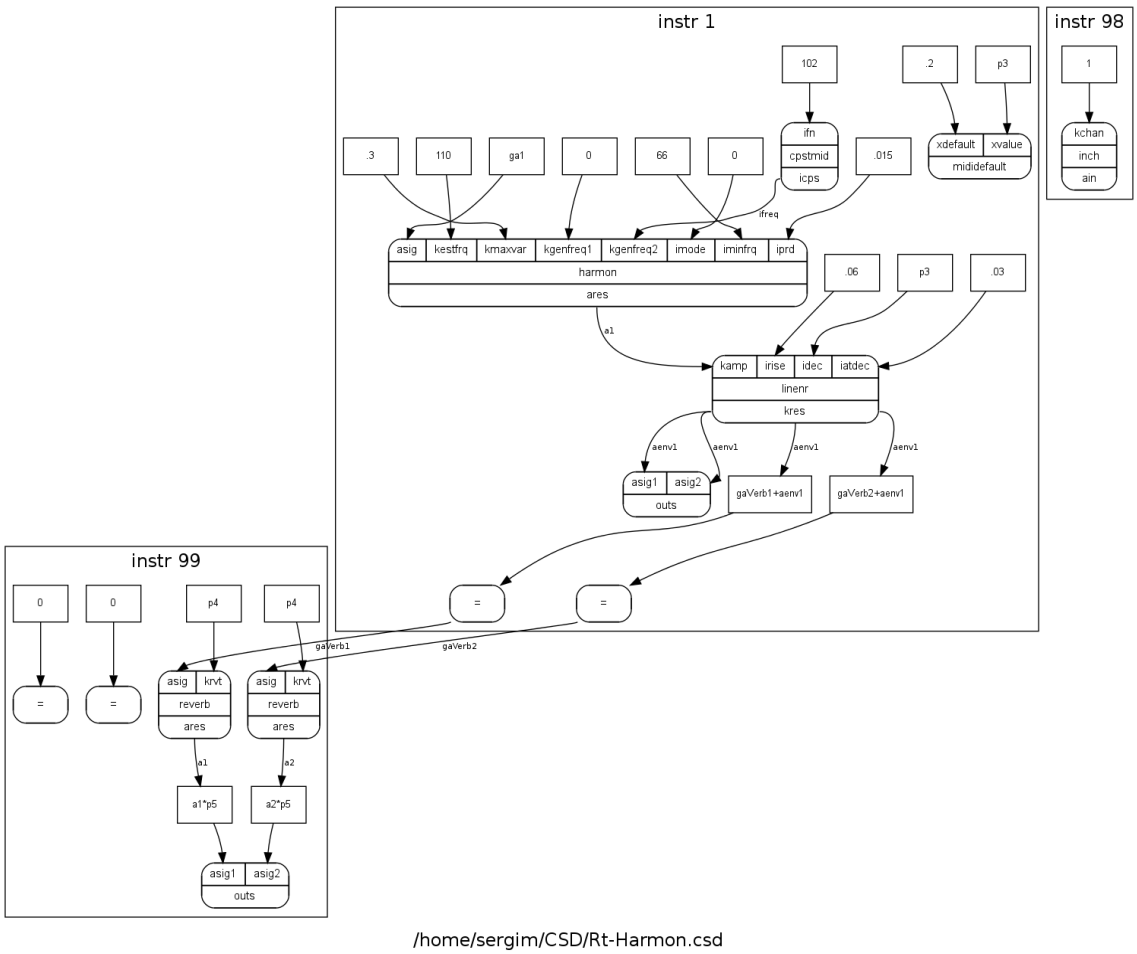


Figura 09: Fluxograma do *Harmonizador em Tempo Real*.

2.5 XY-FM

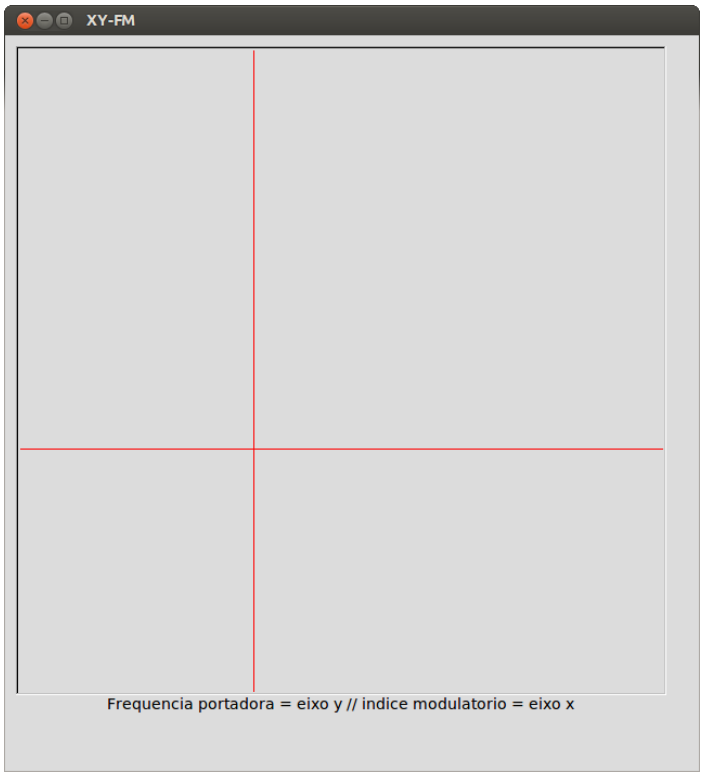


Figura 10: Captura de tela da Interface de XY-FM.

XY/FM é uma interface simples criada para testar algumas possibilidades timbrísticas da síntese FM, funciona com um vetor bidimensional, dois *sliders* simultâneos (*joystick*) no gráfico com eixos X e Y, sendo, o controle da frequência portadora em "Y", e em "X" podemos controlar a amplitude da frequência que modula a frequência portadora, o índice modulatório. A frequência moduladora está definida para ser a mesma da frequência portadora e pode ser alterada apenas dentro do código para o estudo da invenção de John Chowning em 1967, a síntese sonora por modulação de frequência, Puckette (2006. p. 141). Seu código se localiza no ANEXO A em Algoritmo nº 4, *on-line* pela *url* disponibilizada na página 38 e CD-ROM anexo.

2.6 Delay Doppler (Iain McCurdy)

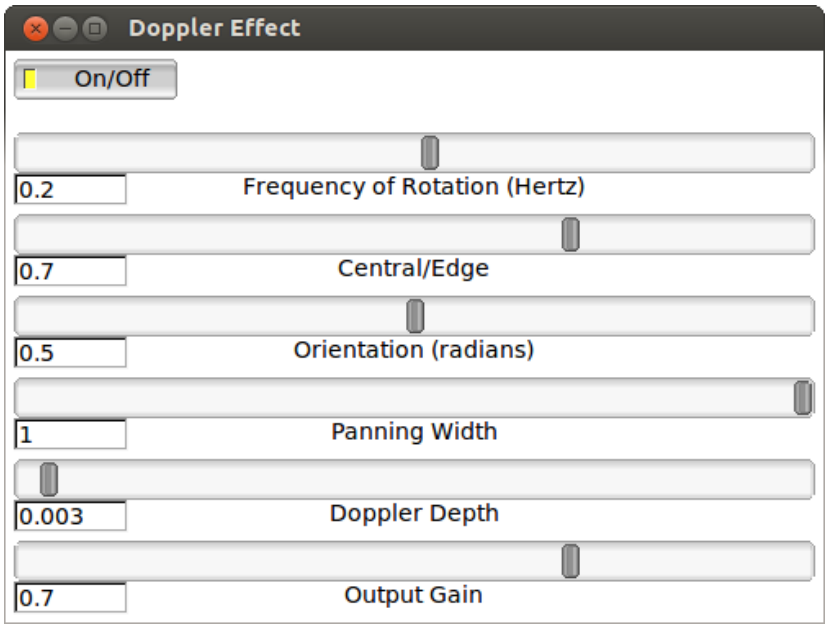


Figura 11: Captura de tela da interface Delay Doppler de Iain McCurdy em *Csound realtime examples* do diretório: Delays.

Este exemplo utiliza três LFOs (oscilação de baixa frequência) para criar o efeito de um movimento de som em círculo à volta do ouvinte. Os três parâmetros controlados por essas LFOs são amplitude, panorama e tempo de atraso. A modulação do tempo de atraso também resulta em uma modulação de campo, que é conhecido como o efeito de Doppler. O importante para este efeito é que todas as três LFOs compartilham o mesmo valor de frequência. Valores negativos de frequência também são permitidos - o que representaria uma mudança na direção da fonte de sons em movimento em torno de nós. Também de importância fundamental é a relação de fase entre os três LFOs pois esta define exatamente onde a fonte de som está em relação ao ouvinte. O LFO panoramizador deve estar em seu ponto mínimo de mudança então a fonte sonora se desloca paralelamente à direção em que o ouvinte está de frente; ou seja, diretamente para a esquerda ou para a direita do ouvinte. O tempo de atraso LFO (pitch modulação / doppler) deve estar em seu ponto mínimo de mudança então a fonte de som se move perpendicularmente à direção em que o ouvinte está de frente; ou seja, diretamente na frente ou atrás do ouvinte. A diferença de fase entre esses dois LFOs é ou 90º ou 270º, dependendo se a fonte de som está se movendo no sentido horário ou sentido anti-horário em torno de nós. Uma modulação de amplitude entra em jogo sempre que não estamos a ouvir a partir do centro do círculo de movimento. Quanto mais

próximo da margem do círculo, maior a quantidade de modulação de amplitude experimentamos. Se a modulação de amplitude é extrema, em seguida, o círculo do movimento da fonte de som deve ser extremamente grande. A fase do LFO para modulação de amplitude é também ajustável (controle deslizante 'Orientação') este define qual borda do círculo que estão mais próximos, por exemplo, superior, inferior, esquerda, direita, etc. É provavelmente sempre melhor, incluir pelo menos uma pequena quantidade de modulação de amplitude porque percebemos os sons diretamente à nossa esquerda ou à direita mais altos, mesmo que permaneçam equidistante de nós. Neste caso, a fase LFO amplitude ('Orientação') deve ser 0,5 (radianos). A forma de onda para todas as três LFOs é uma onda senoidal. isto define o movimento do objeto como sendo circular. Ao usar uma forma de onda diferente, isto seria modelar um movimento não-circular. Há um interessante potencial de experimentação com esta possibilidade.

3. *SAIGAI TO HIGAI* - 災害と被害

De acordo com Coelho e Hida (1999), *Saigai* significa, calamidade, catástrofe, desastre, sinistro, acidente natural. *Higai* significa, dano, prejuízo, estrago causado pelo homem. Portanto a tradução deste título é desastre natural e desastre artificial. A composição é uma narrativa abstrata sobre os eventos ocorridos no Japão em março de 2011 e se divide em 3 partes: o Terremoto (*Jishin* – 地震), o Maremoto (*Tsunami* – 津波) e o Vazamento Radioativo (*Hōsha no More* – 放射の漏れ).

Uma diferença crucial no algoritmo desta composição é o fato de possuir uma sequência de eventos programada que chamamos também de partitura na sintaxe do *Csound*. Esses eventos é que narram aproximadamente a sonoridade possível, porém resumida, dos eventos no Japão.

Primeiramente inicia-se um exercício randômico à duas vozes com escala pentatônica maior que posteriormente se torna menor após o primeiro abalo. A equação que adpta as vibrações sismológicas se aplica na modulação de amplitude de todas as sonoridades da composição. Após os primeiros abalos iniciam sons de impacto de barras de metal que são exemplos de modelagem física nos tutoriais de *Csound* pelo *opcode* **barmodel**. Esses impactos também são aplicados à modulação de amplitude dos sons de escala japonesa causando uma transformação timbrística bastante contrastante, de um som puro (senoidal) a um som próximo ao ruído branco. Após essa primeira parte, são intruduzidos o que metaforicamente chamamos de "ventos uivantes" produzidos com síntese granular do *opcode* **grain** e ruído branco para simular o efeito da invasão da água com o *opcode* **noise**. Nessa parte o acelerômetro não exerce influência, pois, não houveram tremores de terra durante o tsunami. A terceira e última parte se inicia com a simbolização da explosão da usina nuclear, liberando os harmônicos das frequências 137Hz e 131Hz usadas para representar os números atômicos do Césio e do Iodo radioativos. Nessa parte o acelerômetro volta a causar distúrbios na sonoridade residual após a explosão e promove um efeito de som aquático, o simbolismo da contaminação do mar.

Este foi um trabalho de composição algoritmica que envolveu bastante estudo da linguagem *Csound* e formas de síntese anteriores. A narrativa descrita acima sofre distúrbios causados pelos dados do acelerômetro que são similares a dados de sismógrafos, a performance com ele é *ad libttum* e deve representar a imprevisibilidade ou falta de controle

das forças da natureza, a poética das “mãos de Deus” a fazer tremer a terra como desastre natural.

Os *softwares* livres usados são apenas *Csound*, OSC e AndOSC, pois, as composições deste trabalho não se limitam aos programas pesquisados, pelo contrário, durante as composições é que surgem as pesquisas sobre *softwares* Livres para a resolução de problemas composicionais.

A programação para futuras interpretações se encontra no ANEXO A em Algoritmo nº 5, *on-line* pela *url* e *QrCode* abaixo. Também se encontra em mídia CD-ROM anexada a este trabalho.

https://drive.google.com/folderview?id=0Bx_5iQ2UHJJvYmszY1BELWpMZ0E&usp=sharing

3.1 Interface com Acelerômetro

O Algoritmo de *Saigai to Higai* traz a configuração do protocolo OSC de forma a associar o acelerômetro de dispositivos celulares às oscilações de baixa frequência (LFO) que causam perturbações por modulação de amplitude nos eventos previstos da partitura em *Csound*, simbolizando as ondas sísmicas causadoras de todos os desastres.

Na figura 12 temos capturas de tela do aplicativo *TouchOSC* que se encontra disponível tanto para ANDROID como para iOS. Este aplicativo é propietário e foi escolhido pela compatibilidade com mais dispositivos, no entanto, é possivel usar também o *AndOSC* que é gratuito, mas apenas para sistema ANDROID. Para tal troca, é preciso mudar a nomenclatura “/accxyz” na linha do *opcode* “**OSClisten**” por “/acc” dentro do algoritmo.

Assim como no *AndOSC* é importante configurar o aplicativo *TouchOSC* para o endereço TCP-IP e uma porta (*outgoing*), no painel de configurações, como podemos ver na figura 12a. Outra configuração importante, é habilitar o envio dos dados do acelerômetro

como vemos na figura 12b. Para começar o envio desses dados, é preciso tocar em *done*, onde aparecerá um *layout* padrão denominado *simple* que podemos ver na figura 12c.

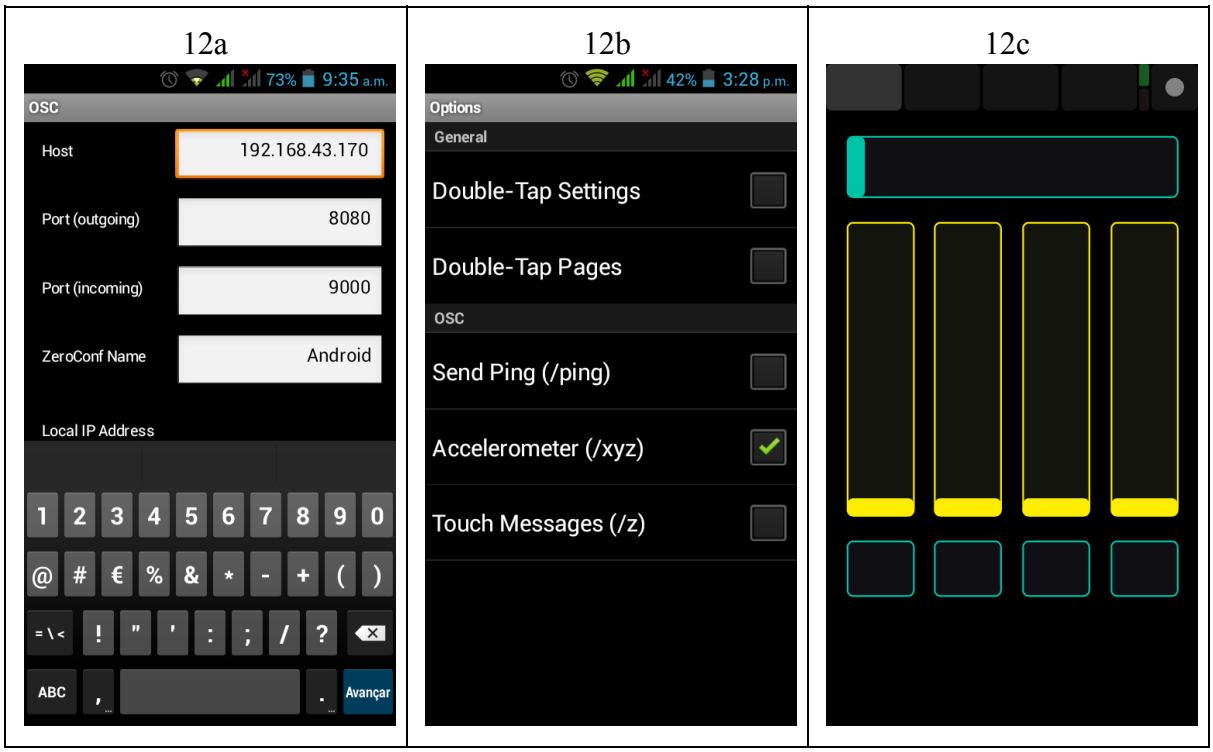


Figura 12: 12a - Captura de tela do painel de configurações do aplicativo *TouchOSC*; 12b - Captura de tela do painel de opções do aplicativo *TouchOSC*; 12c - Captura de tela do *Layout "Simple"* do aplicativo *TouchOSC*.

4. ANS ESTUDO Nº1

Na pesquisa pela composição musical com novas interfaces, encontramos meios para utilizá-las no ato da performance ou no momento da composição. O *software* VirtualANS satisfaz os dois meios, pois, a princípio, realiza a leitura de uma imagem composta anteriormente e também pode ser usado para performance em tempo real.

Num primeiro estudo de suas ferramentas gráficas, descobrimos que a opacidade da imagem está relacionada à intensidade do som, quanto mais transparente, mais piano. Essas ferramentas podemos ver abaixo na figura 13.

Figura 13: Captura de tela das ferramentas gráficas do *software* VirtualANS do desenvolvedor Alexander Zolotov.

No primeiro item temos selecionada a ferramenta para traço livre. Logo abaixo temos o controle de intensidade em *Opacity*. No terceiro item temos diferentes formas de pincéis além de podermos importar qualquer imagem para isso. *Scale* se refere ao tamanho do pincel. *Spacing* regula a distância entre as imagens do pincel no traço, podendo ser contínuo ou espaçado. Logo abaixo, diferentes gradientes (degradê) para criar diferentes envelopes de amplitude como crescendo e decrescendo, e gradientes verticais para diferentes amplitudes dos formantes senoidais.

A ferramenta para desfazer (*undo*) é essencial para o compositor que almeja um objetivo sonoro, pois, quando o resultado obtido não satisfaz o pretendido, este poderá ser refeito até que o resultado seja satisfatório.

Diferentes pincéis nos fornecem diferentes texturas como granulações onde temos o controle absoluto da densidade e tamanho dos grãos. Com a precisão atingida com a ferramenta *zoom,* é possível compor estruturas microtonais. Também é possivel transladar uma progressão harmônica tradicional para o campo micro-tonal ou macro-tonal com a ferramenta *paste+resize* ou alterando o tamanho do pincel quando usamos *make brush*, uma ferramenta essencial para o trabalho de captura de um motivo ou amostra para ser usada como pincel/carimbo, ou ainda, alterando nas configurações do projeto em *settings*, o número de oitavas do âmbito geral.

A composição do estudo nº1 para ANS se restringe a uma expressão livre do ato criador sem nenhuma estruturação antecipada, sendo uma narrativa sonora por modelagem espectral feita com as ferramentas existentes no programa. Logo, é um estudo sobre os resultados sonoros que podem ser obtidos pela síntese direta das possibilidades gráficas de suas ferramentas de trabalho, não envolvendo portanto, processamento de sons acústicos ou a utilização de pincéis que não sejam os padrões do programa.

O aplicativo em questão não possui licença GPL, não sendo um *software* Livre. No entanto, o seu desenvolvedor, Alexander Zolotov cobra pelos seu direitos de autor, apenas em instalações em dispositivos portáteis como *smartphones* e *tablets,* deixando-o livre para instalações em computadores com sistemas MAC OS, Windows e Linux.

A composição possui configurações que são importantes para o registro da obra, ou seja, que complementam a partitura/desenho por definir a gama de oitavas, altura geral e andamento. Abaixo, na figura 14, podemos verificar as configurações para o Estudo Nº1.

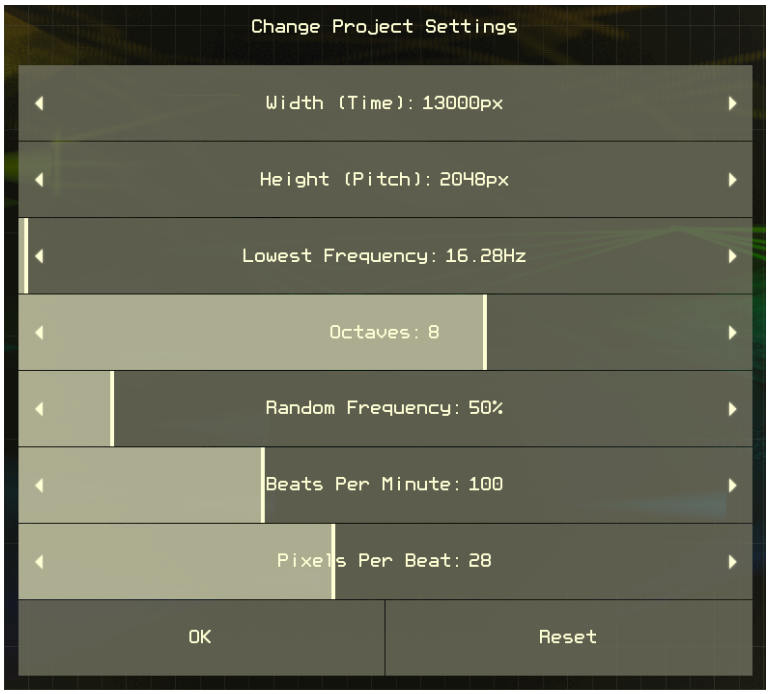


Figura 14: Captura de tela da janela de configurações do projeto: *ANS Estudo N°1* no *software* VirtualANS do desenvolvedor Alexander Zolotov.

A partitura espectrográfica se encontra no ANEXO B, os arquivos digitais como o projeto no formato do VirtualANS, partitura (pdf) e áudio em mp3 disponibilizamos abaixo em endereçamento eletrônico e seu *QrCode*.

https://drive.google.com/folderview?id=0Bx_5iQ2UHJJvRlJaMXlfb2JtTDA&usp=sharing

5. ANS ESTUDO Nº2

A composição de ANS Estudo Nº2 teve sua estrutura pensada antes de ser iniciado os primeiros traços da modelagem espectral da música e a ideia principal era que a imagem da partitura fosse tão importante quanto seu som, ao contrário do primeiro estudo onde a imagem visual é irrelevante, pois, trata-se da materialização direta do som pensado durante o ato de compor/desenhar.

A música é estruturada por dois temas: sujeito e contra-sujeito, onde o sujeito é a sonoridade obtida da imagem do *Kanji*[25] 音楽 (*ongaku*) cujo significado é Música. O contra-sujeito é a sonoridade obtida da imagem do *Kanji* 愛 (*ai*) cujo significado é amor.

A introdução desses temas é como na exposição dos temas de uma fuga, por meio de pergunta e resposta. Após a exposição destes temas, temos um divertimento modulatório construído com a diminuição das imagens dos temas que leva até um desenvolvimento em que o sujeito e o contra-sujeito são recapitulados de forma aumentada. O interessante dessas diminuições e aumentações é que não só o tempo é reduzido ou aumentado, temos também o redimensionamento intervalar das "melodias" produzidas pelas imagens dos *Kanji*.

Após o desenvolvimento, temos uma sessão intermediária onde o ideograma para Música é reduzido à um motivo, e com ele, estruturado o ideograma para amor. Uma simbologia para "amor feito com música". Nessa parte, a sonoridade é de acordes diminutos, pois, o espaçamento entre cada pequeno *Kanji* de música é de 4 (quatro) imagens por oitava, fazendo uma sonoridade de tensão.

Após a sessão intermediária, temos a sessão final onde o ideograma para amor é reduzido à um motivo, e com ele, estruturado o ideograma para música. Uma simbologia para "música feita com amor". Apesar da sonoridade nada convencional, a música começa em Dó e termina em Dó como se fosse música tonal. Além disso, todas as alturas onde se localizam os temas foram pensadas.

Tal como o Estudo Nº1, esta composição possui configurações que definem a região audível que se localiza a partitura/desenho e complementam o registro da obra. Abaixo, na figura 15, as definições do Estudo Nº2.

[25] O *Kanji* (漢字) é o ideograma ("ideia desenhada" ou "ideia escrita"), ou seja, é provido de um significado, sendo de origem chinesa. Biderman (2001) afirma que o ideograma está ligado ao plano do conteúdo e não ao plano da expressão, ou seja, o ideograma é o significado e não o significante na concepção de significado e significante de Saussure (2002).

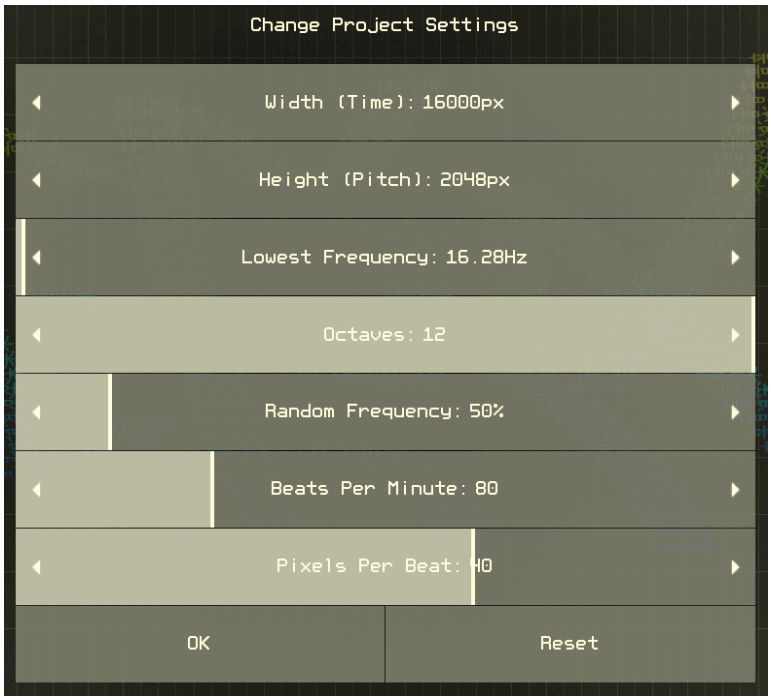


Figura 15: Captura de tela da janela de configurações do projeto: *ANS Estudo Nº2* no *software* VirtualANS do desenvolvedor Alexander Zolotov.

A partitura espectográfica se encontra no ANEXO B. O projeto no formato do VirtualANS, partitura (PDF), os ideogramas para os pincéis e o áudio em mp3 disponibilizamos abaixo pela *url*, seu QrCode e CD-ROM anexo.

https://drive.google.com/folderview?id=0Bx_5iQ2UHJJvfmh1MWNZbGYzZWdaMjRLTGM3WTVncW1SMl94STVHVDQ0RUREbGJEM0RXLUk&usp=sharing

6. CONCLUSÕES

O objetivo principal desse trabalho foi a criação de um conjunto de composições musicais com foco na implementação e uso de controles não convencionais para síntese sonora com alicerce em *Software* Livre, desde o sistema operacional. O conceito de interface aqui empregado adota a definição de Caesar (1997), onde o termo é amplamente utilizado para os diferentes modos de acesso a controles de processos de produção sonora. Implicitamente ficou excluída a utilização de DMIs (*digital musical instruments*) que simulam instrumentos convencionais através da utilização do protocolo MIDI (*Musical Instrument Digital Interface*).

Stravinsky (1996), em seu livro sobre poética musical afirma que a liberdade é conquistada dentro de limites e regras. Pode parecer contraditório, mas se tratando de construção com motivos e princípios, sem regras e limites não é possível uma estruturação. A liberdade absoluta em que não há nenhuma regra ou limite, não oferece um princípio estruturador, causando no compositor uma crise inicial em sua criação.

Um instrumento musical ou uma interface computacional que emitem sons enquanto "tocados" são em si, os limites e regras para a sua produção sonora, é isso que caracteriza o seu timbre, as frequências que emite e sua massa sonora. E ao contrário dos instrumentos acústicos convencionais, com as interfaces de computador podemos delegar à máquina os aspectos mecanizados da performance, livrando o intérprete do peso do virtuosismo e permitindo sonoridades complexas através de outro nível de gestualidade, menos física, mais conceitual.

Portanto este trabalho buscou a tentativa de junção entre improviso musical com interfaces de computador com os princípios de estruturação composicional. Uma mistura de expressão gestual na produção sonora em busca de elementos humanos na música eletrônica e estruturação composicional aplicada à linguagem computacional. Nessa busca pela sonoridade não estocástica da música de computador gerada por sorteios pseudo-aleatórios, a improvisação com interfaces que transformam os sons em tempo real se mostra uma excelente solução para a produção de sons inauditos que sejam expressivos da vontade humana, ao contrário de sons obtidos por resultados de equações.

Um fator de discussão na estética eletroacústica são as diferenças entre música acusmática e música para performance ao vivo. Boulez defende a performance ao vivo, a

música de sala de concerto, aquela que não suprimiu o intérprete e sempre terá uma nova versão/interpretação. Por outro lado, Caesar afirma que a música fixada (gravada) fornece possibilidades de percepções de ordem diferente das que o ouvinte se habituou a experimentar. E ainda neste sentido, a forma de divulgação da música nos dias atuais são em sua maioria por meios digitais em gravações e é onde se encaixa muito bem a música fixada.

Em nossa criação artística aqui exposta, estabelecemos partes definitivas (suporte fixo) até mesmo nas composições com performance ao vivo, pois, a criação de algoritmos computacionais para interfaces que sejam interativas no sentido da máquina para o homem é objetivo que exige mais tempo de maturação com as linguagens de programação ou o apoio de uma equipe de pesquisa, como Pierre Boulez obteve auxílio do IRCAM na composição de *Anthèmes 2*. A presença de pré-estabelecimento em composições, apontam que o compositor musical determina a estrutura de sua música, mesmo contendo informações para liberdade total ao intérprete (*ad libitum*), fazendo com que o conceito do criador estruture sua arte mesmo ao delegar ao intérprete liberdade para fazer seu próprio acabamento. Observamos também, que o ato criativo do compositor se inicia desde a composição algoritmica para criação das interfaces alternativas para o gesto até a mixagem e masterização no caso das partes fixas e composições acusmáticas. A criação está presente também na confecção das partituras que são necessárias para performance em palco no caso das composições com performance ao vivo.

A composição não linear obtida por programa multi-pista estabelece um meio intermediário entre estruturação composicional e a improvisação encontrada no *Jazz*, pois, se torna possível estruturar uma composição a partir de frases obtidas de improvisos em gravações e regravações até que seja obtido um resultado satisfatório. Este foi o meio encontrado na composição de *Caverna da Medusa* que por final se definiu fixada no sentido de fixidez de Menezes (1998) onde a variedade de linhas e complexidade causam uma percepção diferente a cada audição e o rítmo não marcado da composição evita a percepção de rigidez temporal do pulso da máquina no momento de execução.

Das composições apresentadas neste trabalho, somente *Vênus e Marte* e *Saigai to Higai* possuem performance ao vivo com instrumentos alternativos e é importante alertar que para a performance com o giroscópio em *Vênus e Marte*, não devem haver objetos metálicos próximos ao performer, por causarem distúrbios magnéticos já que o sensor de Azimute é como uma bússola. Na estréia da música no SEMPEM XIII é possível perceber que ocorreram

inconsistências causadas pelos metais presentes no interior do piano que estava próximo no palco.

O estudo de uma obra para a sua execução e as repetições no ato de memorizar para reproduzir o conteúdo de forma expressiva, são trabalhos que exigem tempo de estudo para o intérprete. *Vênus e Marte*, a exemplo, demanda esse tempo para sua interpretação. Inversamente a isso, *Saigai to Higai* envolveu mais pesquisa de formas de síntese e estruturação algorítmica com a linguagem *Csound* onde suas sonoridades já estabelecidas recebem as interferências relativas aos abalos sísmicos que podem ser produzidos de forma absolutamente livre pelo intérprete no ato da performance com um acelerômetro de três eixos. Porém, ainda apresenta necessidade de calibragem do acelerômetro de acordo com o objeto ou ser vivo que irá fazer o "toque de Deus" ou força da natureza a causar interferências na narrativa do seu algoritmo.

A imprevisibilidade está presente em ambas composições com performance ao vivo e a diferença entre as duas é que uma evita o imprevisível ao máximo (*Vênus e Marte*) e a outra faz uso da imprevisibilidade (*Saigai to Higai*).

Sobre o *software* VirtualANS, este satisfaz a vontade do compositor por estruturação musical em meio gráfico como compor em uma partitura, fornece uma forma de edição não linear e ainda engloba todas as possibilidades eletroacústicas como a síntese e a transformação de sons concretos, pois, possui a opção de importar gráficos ou áudio que podem ser gerados externamente. O programa é um intermediador entre imagem e som, dando a liberdade para o compositor estruturar sua obra pelo som ou pela imagem e obter como resultado final essas duas vias de registro. A composição do Estudo Nº1 e Estudo Nº2 para VirtualANS são as primeiras experiências com o programa e se diferenciam drasticamente já que a primeira é uma expressão livre do ato criador sem a menor estruturação antecipada, ao contrário da segunda, onde a estruturação pela imagem foi concebida antecipadamente.

Por fim, o casamento entre conhecimento musical e tecnológico gratificam os anseios do compositor contemporâneo em trazer o som do mundo das ideias para o mundo físico e poder comunicar sua expressão a todas as pessoas. Diante da infinidade das possibilidades da composição musical moderna podemos sentir independência na produção sonora e apreciar a obra no ato de seu talhamento. Esta pesquisa não se encerra aqui e novas composições serão criadas à luz deste estudo.

REFERÊNCIAS

ALMEIDA, Anselmo Guerra de. **Sistemas Musicais Interativos: Metáforas e Métodos**. In: XXI Congresso da Associação Nacional de Pesquisa e Pós-Graduação em Música - Uberlândia, 2011. Disponível em <https://www.academia.edu/870892/Sistemas_Musicais_Interativos_Metáforas_e_Métodos>. Acesso em 5 de setembro de 2014.

BIDERMAN, Maria Tereza Camargo. **As Ciências do Léxico.** In: OLIVEIRA, Ana Maria Pinto Pires de & ISQUERDO, Aparecida Negri (orgs.). **As Ciências do Léxico: Lexicologia, Lexicografia, Terminologia.** 2ª edição. Campo Grande: UFMS, 2001. p.13-22.

BOULEZ, Pierre. **A Música Hoje**. São Paulo: Editora Perspectiva, 3ª edição, 1986.

BOULEZ, Pierre. Conferénce de Pierre Boulez sur Anthèmes 2. In: GOLDMAN, Jonathan. **Understanding Pierre Boulez's Anthèmes [1991]: Creating a Labyrinth out of Another Labyrinth.** Tese de doutoramento. Montréal: Faculty of music - Université de Montréal, 2001.

CAESAR, Rodolfo. **Novas interfaces e a produção eletroacústica.** In: Anais do SBCM, Brasíllia, 1997.

COELHO, Jaime e HIDA, Yoshifumi. **Dicionário Universal Japonês-Português.** 3ª ed. Japão: Shogakukan, 1999.

FIGUEIRÓ, Cristiano S. **Composição Interativa - Estratégias e Relatos de Processos.** Goiânia: UFG, 2005. (Dissertação).

FREED, Adrian e SCHMEDER, Andy. **Features and Future of Open Sound Control**, version 1.1 for NIME. Disponível em <http://cnmat.berkeley.edu/system/files/attachments/Nime09OSCfinal.pdf>. Acesso em 5 de setembro de 2014.

GUICHENEY, Paulo C. N. **Dois Percursos Composicionais em Música Eletroacústica**. Goiânia: UFG, 2006. (Dissertação).

KREICHI, Stanislav. **The ANS Synthesizer: Composing on a Photoelectronic Instrument.** Disponível em < http://theremin.ru/archive/ans.htm > Acesso em 5 de setembro de 2014.

KREIDLER, Johannes. **Programming Electronic Music in Pd**. 2009. Disponível em <http://www.pd-tutorial.com/english/index.html>. Acesso em 5 de setembro de 2014.

LEE, Allan S. C. **Granular Synthesis in Csound.** In: The Csound Book, MIT Press, Massachussets, 2000.

LIMA, Gabriel Rimoldi de. **Modelagem Interativa Aplicada à Síntese e Espacialização no Domínio Microtemporal**. Campinas: UNICAMP, 2013. (Dissertação)

MACHOVER, Tod. Principal Investigator. **Hyperinstruments - A Progress Report 1987 - 1991**. MIT Media Laboratory. January, 1992.
MCCURDY, Iain. **Csound Realtime Examples**. Disponível em <http://iainmccurdy.org/csound.html>. Acesso em 5 de setembro de 2014.

MENEZES, Flo. **Atualidade estética da música eletroacústica**. São Paulo: Editora da UNESP, 1998.

MIRANDA, Eduardo R.; WANDERLEY, Marcelo M. **New Digital Musical Instruments: Control and Interaction Beyond the Keyboard**. AR Editions, Middleton, Wisconsin, 2006. Disponível em <http://books.google.com.br/books?id=CGEwXZ7hcIoC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acesso em 13 de Março de 2014.

PUCKETTE, Miller. **The Theory and Technique of Electronic Music**. Hackensack, N.J.: World Scientific Publishing Co., 2006.

RAY, Sonia. **Editorial.** Música Hodie, Goiânia, v. 11, n. 2, p. 5-6, 2011.

SAUSSURE, Ferdinand de. **Curso de Linguística Geral.** 22ª edição. São Paulo: Cultrix, 2002.

SCHAEFFER, Pierre. **Tratado dos objetos sonoros**. Edunb, Brasília,1994.

SMITH, Brett. **A Quick Guide to GPLv3.** 2007. Disponível em <http://www.gnu.org/licenses/quick-guide-gplv3.pdf >. Acesso em 5 de setembro de 2014.

STRAVINSKY, Igor. **Poética Musical em Seis Lições.** Rio de Janeiro: Editora Jorge Zahar, 1ª edição, 1996.

TRALDI, Cesar A. **Percussão e Interatividade PRISMA: Um Modelo de Espaço Instrumento Auto-Organizado.** Campinas: UNICAMP, 2009. (Tese)

VERCOE, Barry. **The Canonical Csound Reference Manual**, Version 6.04. Disponível em <http://csound.github.io/docs/manual/index.html>, MIT Media Lab. 2014. Acesso em 5 de setembro de 2014.

ANEXO A

Algorítmos

Algoritmo nº 1

Sintetizador por Randomização de Harmônicos

```
;----------------- Sintetizador por Randomizacao de harmonicos ------- by: Sergim Veiga
<CsoundSynthesizer>
<CsOptions>
-d            ;  \
-odac          ;  \
</CsOptions>    ;   <---- Cabeçalho com saída DAC 48kHz, stereo e taxa de cotrole 1 por 1.
<CsInstruments>;  /
sr = 48000    ;  /
ksmps = 1
nchnls = 2

FLcolor   255,   255,   255 ; <---- determina cor branca para a janela
FLpanel  "Sintetizador por Randomizacao de harmonicos",810,400,0,0; <-- Abre um painel

idvol     FLvalue   "0-5000",      70,       30,    725,  20
idgama    FLvalue   "1-512",       70,       30,    725,  80   ;  <--- criam monitoramento
idfrgr    FLvalue   "1-200Hz",        70,       30,    725,  140;        de valores
idfrfu    FLvalue   "1-12000Hz",      70,       30,    725,  200
idT60     FLvalue   ".01-20s",        70,       30,    725,  260

gk1,ih1   FLslider  "Volume", 1,5000,-1,25, idvol, 700,30, 20,20
gk2,ih2   FLslider  "Gama de harmonicos", 1,512,-1,25, idgama, 700,30, 20,80
gk3,ih3   FLslider  "Frequencia das randomizacoes", 1,200,-1,25, idfrgr, 700,30, 20,140
gk4,ih4   FLslider  "Frequencia fundamental", 1,12000,-1,25, idfrfu, 700,30, 20,200
gk5,ih5   FLslider  "Tempo de reverberacao", .01,20,-1,25, idT60, 700,30, 20,260

      FLsetVal_i      2000, ih1
      FLsetVal_i      12,   ih2
      FLsetVal_i      8,    ih3   ;  <------ determinam valores iniciais (default)
      FLsetVal_i      440,  ih4
      FLsetVal_i      3,    ih5

ihShow    FLbox "Por: Sergio de Alencastro Veiga Filho",7,10,20,500,30,150,340

      FLrun   ; <----- Fecha o painel e determina abrí-lo.

ga1    init  0    ;   <----  Variável global para o sinal de Reverberação

instr 1
kamp = gk1

k1 randomh 1, gk2, gk3    ;   <----  Randomizador

kcps = gk4*int(k1) ;  <--- transformação em números inteiros que multiplicam a frequência

a1 oscili kamp, kcps, 1  ;   <----  Oscilador

a2 linseg 0,.004,1,p3-.008,1,.004,0
   outs a1*a2,a1*a2
ga1 = ga1+(a1*a2)
endin
instr 99 ; <-- instrumento para a reverberação

a1 reverb ga1,gk5 ;   <---- gk5 determina o T60 (tempo de atenuação em 60dB)
   outs a1*.2,a1*.2
ga1 = 0
endin
</CsInstruments>
<CsScore>
f1 0 16384 10 1    ;  <----  Função senoidal
i99 0 3600     ;   <---- duração de 3600 segundos (1 hora) para a reverberação
i1 0 3600      ;   <---- duração de 3600 segundos (1 hora) para o instrumento
e
</CsScore>
</CsoundSynthesizer>
```

Algoritmo nº 2

Theremin Giroscópico

```
; Written by Iain McCurdy, 2006
; OSC Implemented by Sergim Veiga, 2013
<CsoundSynthesizer>
<CsOptions>
-d
-odac -b64
</CsOptions>
<CsInstruments>

;DEMONSTRATION OF THE FLpanel OPCODE
;HORIZONTAL CLICK-AND-DRAGGING CONTROLS THE FREQUENCY OF THE OSCILLATOR
;VERTICAL CLICK-AND-DRAGGING CONTROLS THE AMPLITUDE OF THE OSCILLATOR

sr	=	48000
ksmps	=	2
nchnls	=	2

giosc1 OSCinit 8080

;FLTK INTERFACE
CODE;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;
FLcolor	255, 255, 255, 0, 0, 0
;	OPCODE	LABEL		| WIDTH | HEIGHT | X | Y
	FLpanel	"Theremin - Giroscópico",	1225,	700,	0,	0

;gkfreqx,gkampy,ihandlex,ihandley FLjoy LABEL, IMINX, IMAXX, IMINY, IMAXY, IEXPX, IEXPY,
IDISPX, IDISPY, IWIDTH, IHEIGHT, IX, IY
gkoctx, gky, ihandlex, ihandley	FLjoy " ",	7,	11,	0,	1024,	0,	0,
-1,	-1,	1200,	150,	13, 100

;		WIDTH | HEIGHT | X	| Y
iddlytim	FLvalue	" ",	60,	20,	5,	325
iddlyFB		FLvalue	" ",	60,	20,	5,	375
iddlymix	FLvalue	" ",	60,	20,	5,	425
idRoomSize	FLvalue	" ",	60,	20,	5,	475
idHFDamp	FLvalue	" ",	60,	20,	5,	525
idrvbmix	FLvalue	" ",	60,	20,	5,	575

idmodfrq	FLvalue	" ",	60,	20,	505,	325
idvibdep	FLvalue	" ",	60,	20,	505,	375
idtrmdep	FLvalue	" ",	60,	20,	505,	425
idoctport	FLvalue	" ",	60,	20,	505,	475
iddampport	FLvalue	" ",	60,	20,	505,	525
idoutamp	FLvalue	" ",	60,	20,	505,	575
idporttime	FLvalue	" ",	60,	20,	505,	625

;				MIN	|	MAX	| EXP | TYPE |	DISP	|	WIDTH | HEIGHT | X	| Y
gkdlytim, ihdlytim FLslider "Delay Time",	.0001,	2,	0,	23,	iddlytim,	490,
25,	5, 300
gkdlyFB,	ihdlyFB	FLslider "Delay Feedback", 0,	1,	0,	23,	iddlyFB,	490,
25,	5, 350
gkdlymix, ihdlymix FLslider "Delay Mix",	0,	1,	0,	23,	iddlymix,	490,
25,	5, 400
gkRoomSize, ihRoomSize	FLslider "Room Size",	0,	1,	0,	23,	idRoomSize,
490,	25,	5, 450
gkHFDamp, ihHFDamp FLslider "H.F. Diff,",	0,	1,	0,	23,	idHFDamp,	490,
25,	5, 500
gkrvbmix, ihrvbmix FLslider "Reverb Mix",	0,	1,	0,	23,	idrvbmix,	490,
25,	5, 550

gkmodfrq, ihmodfrq FLslider "Modulation Frequency", 0,	30,	0,	23,	idmodfrq,
490,	25,	505, 300
gkvibdep, ihvibdep FLslider "Vibrato Depth",	0,	1,	0,	23,	idvibdep,	490,
25,	505, 350
gktrmdep, ihtrmdep FLslider "Tremolo Depth",	0,	1,	0,	23,	idtrmdep,	490,
25,	505, 400
gkoctport, ihoctport	FLslider "Pitch Portamento",	0,	1,	0,	23,	idoctport,
```

```
490,     25,   505, 450
gkampport, ihampport  FLslider "Amplitude Portamento", 0,        1,     0,    23,
idampport,  490,     25,   505, 500
gkoutamp, ihoutamp FLslider "Output Amplitude",  0,        1,     0,    23,  idoutamp,
490,     25,   505, 550
gkporttime, ihporttime   FLslider "Portamento Time",0,        1,     0,    23,  idporttime,
490,     25,   505, 600

;INITIALISE VALUATORS    VALUE | HANDLE
      FLsetVal_i   .6,   ihdlytim
      FLsetVal_i   .3,   ihdlyFB
      FLsetVal_i   .2,   ihdlymix
      FLsetVal_i   .9,   ihRoomSize
      FLsetVal_i   .5,   ihHFDamp
      FLsetVal_i   .1,   ihrvbmix

      FLsetVal_i   5.5,  ihmodfrq
      FLsetVal_i   .05,  ihvibdep
      FLsetVal_i   0, ihtrmdep
      FLsetVal_i   .17,  ihoctport
      FLsetVal_i   .2,   ihampport
      FLsetVal_i   .5,   ihoutamp
      FLsetVal_i   .1,   ihporttime
;     FLcolor ired, igreen, iblue
      FLcolor 255,   255,    255

;BORDERS           TYPE | FONT | SIZE | WIDTH | HEIGHT | X | Y
ih        FLbox     "C ", 6,       9,     15,     25,     100,       1,  0
ih        FLbox     "D ", 6,       9,     15,     25,     100,    50+1,  0
ih        FLbox     "E ", 6,       9,     15,     25,     100,   100+1,  0
ih        FLbox     "F ", 6,       9,     15,     25,     100,   125+1,  0
ih        FLbox     "G ", 6,       9,     15,     25,     100,   175+1,  0
ih        FLbox     "A ", 6,       9,     15,     25,     100,   225+1,  0
ih        FLbox     "B ", 6,       9,     15,     25,     100,   275+1,  0
ih        FLbox     "C ", 6,       9,     15,     25,     100,   300+1,  0
ih        FLbox     "D ", 6,       9,     15,     25,     100,   350+1,  0
ih        FLbox     "E ", 6,       9,     15,     25,     100,   400+1,  0
ih        FLbox     "F ", 6,       9,     15,     25,     100,   425+1,  0
ih        FLbox     "G ", 6,       9,     15,     25,     100,   475+1,  0
ih        FLbox     "A ", 6,       9,     15,     25,     100,   525+1,  0
ih        FLbox     "B ", 6,       9,     15,     25,     100,   575+1,  0
ih        FLbox     "C ", 6,       9,     15,     25,     100,   600+1,  0
ih        FLbox     "D ", 6,       9,     15,     25,     100,   650+1,  0
ih        FLbox     "E ", 6,       9,     15,     25,     100,   700+1,  0
ih        FLbox     "F ", 6,       9,     15,     25,     100,   725+1,  0
ih        FLbox     "G ", 6,       9,     15,     25,     100,   775+1,  0
ih        FLbox     "A ", 6,       9,     15,     25,     100,   825+1,  0
ih        FLbox     "B ", 6,       9,     15,     25,     100,   875+1,  0
ih        FLbox     "C ", 6,       9,     15,     25,     100,   900+1,  0
ih        FLbox     "D ", 6,       9,     15,     25,     100,   950+1,  0
ih        FLbox     "E ", 6,       9,     15,     25,     100,  1000+1,  0
ih        FLbox     "F ", 6,       9,     15,     25,     100,  1025+1,  0
ih        FLbox     "G ", 6,       9,     15,     25,     100,  1075+1,  0
ih        FLbox     "A ", 6,       9,     15,     25,     100,  1125+1,  0
ih        FLbox     "B ", 6,       9,     15,     25,     100,  1175+1,  0
ih        FLbox     "C ", 6,       9,     15,     25,     100,  1200+1,  0

;     FLcolor ired, igreen, iblue
      FLcolor 0,      0,      0

ih        FLbox     "C#", 6,       9,     15,     25,     60,    25+1,  0
ih        FLbox     "D#", 6,       9,     15,     25,     60,    75+1,  0
ih        FLbox     "F#", 6,       9,     15,     25,     60,   150+1,  0
ih        FLbox     "G#", 6,       9,     15,     25,     60,   200+1,  0
ih        FLbox     "A#", 6,       9,     15,     25,     60,   250+1,  0
ih        FLbox     "C#", 6,       9,     15,     25,     60,   325+1,  0
ih        FLbox     "D#", 6,       9,     15,     25,     60,   375+1,  0
```

```
ih        FLbox     "F#", 6,       9,    15,    25,     60,   450+1,  0
ih        FLbox     "G#", 6,       9,    15,    25,     60,   500+1,  0
ih        FLbox     "A#", 6,       9,    15,    25,     60,   550+1,  0
ih        FLbox     "C#", 6,       9,    15,    25,     60,   625+1,  0
ih        FLbox     "D#", 6,       9,    15,    25,     60,   675+1,  0
ih        FLbox     "F#", 6,       9,    15,    25,     60,   750+1,  0
ih        FLbox     "G#", 6,       9,    15,    25,     60,   800+1,  0
ih        FLbox     "A#", 6,       9,    15,    25,     60,   850+1,  0
ih        FLbox     "C#", 6,       9,    15,    25,     60,   925+1,  0
ih        FLbox     "D#", 6,       9,    15,    25,     60,   975+1,  0
ih        FLbox     "F#", 6,       9,    15,    25,     60,  1050+1,  0
ih        FLbox     "G#", 6,       9,    15,    25,     60,  1100+1,  0
ih        FLbox     "A#", 6,       9,    15,    25,     60,  1150+1,  0

      FLpanel_end ;END OF PANEL CONTENTS
      FLrun     ;RUN THE WIDGET THREAD!
      zakinit   1,1

      instr 1
kX init 0                                                ;\
kY init 0                                                 ;\
kZ init 0                                                  ;  <--- linhas descritas no artigo
                                                          ;/
kans1 OSClisten giosc1, "/ori", "fff", kX,kY,kZ    ;/
;                                                 ;/
gkoctx = 7+(kX/90)                              ;/
gky = kZ*12

iporttime=  .05
kporttimelinseg0,.001,iporttime,1,iporttime
kporttime=  kporttime * gkampport

iporttime2  =  .2
kporttime2  linseg0,.001,iporttime2,1,iporttime2
kporttime2  =  kporttime2 * gkoctport

ifn   =  1
kamp      table gky, 2
kamp      portk kamp, kporttime
;koct     linetogkoctx, kporttime2
koct      portk gkoctx, kporttime2
kmoddepthtable gky, 3
kmoddepthportk kmoddepth, kporttime
kmod      oscilikmoddepth, gkmodfrq, 99
koct      =  koct + (kmod*gkvibdep)

aamp      interpkamp
koutamp      portk gkoutamp, kporttime

ktrm      osciligktrmdep*.5*kmoddepth, gkmodfrq, 99
ktrm      =  ktrm+.5

asig      osciliaamp*20000*ktrm*koutamp, cpsoct(koct), ifn
      zawm  asig, 1
      endin

      instr 2  ; PING PONG DELAY
kporttimelinseg0,.001,1,1,1
kporttime=  kporttime*gkporttime
imaxdelay=  2     ;MAXIMUM DELAY TIME

kdlytim      portk gkdlytim, kporttime
adlytim      interpkdlytim

ain   zar1

;LEFT CHANNEL OFFSETTING DELAY (NO FEEDBACK!)
```

```
aBuffer      delayr imaxdelay*.5
aLeftOffset  deltap3  adlytim*.5
      delayw ain

;LEFT CHANNEL DELAY WITH FEEDBACK
aFBsigL      init   0
aBuffer      delayr imaxdelay
aDlySigL  deltap3   adlytim
      delayw aLeftOffset+aFBsigL
aFBsigL      =   aDlySigL * gkdlyFB

;RIGHT CHANNEL DELAY WITH FEEDBACK
aFBsigR      init   0
aBuffer      delayr imaxdelay
aDlySigR  deltap3   adlytim
      delayw ain+aFBsigR
aFBsigR      =   aDlySigR * gkdlyFB

amixL     ntrpol ain, aDlySigL+aLeftOffset,  gkdlymix
amixR     ntrpol ain, aDlySigR,        gkdlymix

      denorm    amixL, amixR ;DENORMALIZE BOTH CHANNELS OF AUDIO SIGNAL
arvbL, arvbR    freeverb  amixL, amixR, gkRoomSize, gkHFDamp , sr
amixL     ntrpol    amixL, arvbL, gkrvbmix   ;CREATE A DRY/WET MIX BETWEEN THE DRY AND THE
REVERBERATED SIGNAL
amixR     ntrpol    amixR, arvbR, gkrvbmix   ;CREATE A DRY/WET MIX BETWEEN THE DRY AND THE
REVERBERATED SIGNAL
      outs   amixL, amixR ;SEND DELAY OUTPUT SIGNALS TO THE SPEAKERS
      zacl   1,1
      endin

</CsInstruments>
<CsScore>
;f 1 0 129 10 1 .1 .05 .025 .0125 .00625 .003125     ;FUNCTION TABLE THAT DEFINES A SINGLE
CYCLE OF A SINE WAVE
;f 1 0 129 10 .5 .8 1 .1 .05 .025 .0125 .00625 .003125     ;FUNCTION TABLE THAT DEFINES A
SINGLE CYCLE OF A SINE WAVE
;f 1 0 129 10 .5 1 .1    ;FUNCTION TABLE THAT DEFINES A SINGLE CYCLE OF A SINE WAVE
f 1 0 129 10 1 .1 ;.05 .025 .0125 .00625 .003125     ;FUNCTION TABLE THAT DEFINES A SINGLE
CYCLE OF A SINE WAVE
f 2 0 1024 7 0 40 0 200 1 [1024-240] 1
f 3 0 1024 7 0 260 0 [1024-260] 1

f99 0 129 10 1

i 1 0 3600     ;INSTRUMENT 1 WILL PLAY A NOTE FOR 1 HOUR
i 2 0 3600     ;INSTRUMENT 2 WILL PLAY A NOTE FOR 1 HOUR
</CsScore>
</CsoundSynthesizer>
```

Algoritmo nº 3

Harmonizador em Tempo Real

```
;------------------------ Harmonizador em tempo real --------------------- By: Sergim Veiga
<CsoundSynthesizer>
<CsOptions>
-d --nodisplays -m0 --sched  ; <--- omite displays (melhor para tempo real)
-+rtmidi=portmidi -M0    ; <--- seleciona entrada MIDI número 1
-+rtaudio=JACK  ; <--- conecta ao servidor de áudio Jack (melhor para tempo real em linux)
-+jack_client=HarmonRT  ; <--- dá nome ao que aparece no Jack
-iadc:system:capture_   ; <--- seleciona para a entrada, o input padrão do jack
;-iadc:vlc_1885:out_  ; <--- seleciona para a entrada, a saída de um player (VLC-jack-plugin)
-odac:system:playback_     ;  <-- seleciona para a saída, o output da placa padrão
</CsOptions>
<CsInstruments>
sr      =       48000
ksmps   =       32
nchnls  =       2

ga1init  0
ga2init  0   ;    <---   variáveis de sinal para entrada estéreo
gaVerb1  init  0
gaVerb2  init  0   ;    <---  variáveis de sinal para reverberação

instr 1
mididefault .2, p3  ;  <-- padrão para envelope de nota midi
ifreq cpstmid 102   ;   <------   converte em ciclos por segundo, a tabela n°102 abaixo
a1 harmon ga1,110,.3,0, ifreq, 0, 66, .015
aenv1 linenr a1,.06,p3,.03  ;
   outs aenv1,aenv1
gaVerb1 = gaVerb1 + aenv1
gaVerb2 = gaVerb2 + aenv1
   endin

   instr 98
ga1 inch 1    ;     <---   entrada de sinal, canal 1
   endin
;-------------------------------
instr  99
a1 reverb gaVerb1, p4
a2 reverb gaVerb2, p4
   outs  a1*p5,a2*p5
gaVerb1   = 0
gaVerb2   = 0
endin
;-------------------------------
</CsInstruments>
<CsScore>
i1 0 10800
i98 0 10800
i99 0 10800 2 .05
     ;----------  Abaixo, tabelas para diferentes afinações, micro-afinações e macro-afinações

; Temperamento igual - Pelas razões aproximadas ou pelas equações (mais preciso)
f1 0 64 -2  12 2  27.5  33  1  1.05946 1.12246  1.189207 1.25992 1.33483 1.41421 1.498307
1.587401 1.68179 1.78179 1.88774
f101 0 64 -2  12 2  27.5  33 1 [2^(1/12)] [2^(2/12)] [2^(3/12)] [2^(4/12)] [2^(5/12)]
[2^(6/12)] [2^(7/12)] [2^(8/12)] [2^(9/12)] [2^(10/12)] [2^(11/12)]

; Afinacao justa em C - Pelas razões ou pelas equações e frações (mais preciso)
f2 0 64 -2  12 2  32.70319  36  1  1.066666  1.125  1.2 1.25 1.333333 1.4 1.5 1.6 1.66666
1.8 1.875
;                          |_--- sendo C o terceiro semitom de 12 a partir de A 440 na tecla
72
f102 0 64 -2 12 2 [440*2^(3/12)] 72 1 [16/15] [9/8] [6/5] [5/4] [4/3] [7/5] [3/2] [8/5]
[5/3] [9/5] [15/8]

; Afinacao Pitagorica - Pelas razões corretas e pelas frações (mesma precisão)
f3 0 64 -2  12 2  [440*2^(3/12)] 72  1  1.06787109375  1.125  1.20135498046875 1.265625
1.35152435302734375 1.423828125 1.5 1.601806640625 1.6875 1.802032470703125 1.8984375
f103 0 64 -2  12 2  [440*2^(3/12)] 72  1  [2187/2048]  [9/8]  [19683/16384] [81/64]
```

```
[177147/131072] [729/512] [3/2] [6561/4096] [27/16] [59049/32768] [243/128]

; Micro afinação por sextos de tom ou 36 notas por oitava - Pelas equações
f36 0 64 -2 36 2 440 69 1 [2^(1/36)] [2^(2/36)] [2^(3/36)] [2^(4/36)] [2^(5/36)] [2^(6/36)]
[2^(7/36)] [2^(8/36)] [2^(9/36)] [2^(10/36)] [2^(11/36)] [2^(12/36)] [2^(13/36)] [2^(14/36)]
[2^(15/36)] [2^(16/36)] [2^(17/36)] [2^(18/36)] [2^(19/36)] [2^(20/36)] [2^(21/36)]
[2^(22/36)] [2^(23/36)] [2^(24/36)] [2^(25/36)] [2^(26/36)] [2^(27/36)] [2^(28/36)]
[2^(29/36)] [2^(30/36)] [2^(31/36)] [2^(32/36)] [2^(33/36)] [2^(34/36)] [2^(35/36)]

; Temperamento igual para Pi - pelas razões - com calculadora e copie e cole
f4 0 64 -2  12 3.1415926535  1  72  1  1.100092378963586982982228646977 4
1.210203242253764275966030761759 1.331335363800389712797534917948 8
1.464591887561523263020142527261 6 1.611186373798326425953803703144 1
1.772453850905516027298167483337 1 1.949862973445820038456833085488 3
2.145029397111025600077444100934 7 2.359730492414696887578474464513 8
2.595921531113400447042449190421 7 2.855753492765337881677360126523 3

; Cromatico por Simetrico -- começa com tom - pelas razões
f5 0 64 -2  12 2.828427124746190097603377448407 6  1  72  1
1.122462048309372981433533049678 7  1.189207115002721066717499970559 7
1.334839854170034364830831881182 9 1.414213562373095048801688724207 7
1.587401051968199474751705639269 4 1.681792830507429086062250952462 9
1.887748625363386993283826313330 3 2 2.244924096618745962867066099351 1
2.378414230005442133434999941112 7 2.669679708340068729661663762358 4

; Cromatico por Simetrico -- começa com semi-tom - pelas razões
f6 0 64 -2  12 2.828427124746190097603377448407 6  27.5  21  1
1.059463094359295264561825294946 1  1.189207115002721066717499970559 7
1.259921049894873164767210607277 1 1.414213562373095048801688724207 7
1.498307076876681498799280732027 4 1.681792830507429086062250952462 9
1.781797436280678609480452411176 9 2 2.118926188718590529123650589886 3
2.378414230005442133434999941112 7 2.519842099789746329534421214547 1

; Harmonicos para oitava 3 - pelas razões
f7 0 64 -2  12 3 1 60 1 1.125 1.25 1.333333 1.5 1.666666 1.875 2 2.125 2.25 2.5 2.666666

; Harmonicos do 16 ao 32 - pelas razões
f8 0 64 -2  16 2 27.5 21 1 1.0625 1.125 1.1875 1.25 1.3125 1.375 1.4375 1.5 1.5625 1.625
1.6875 1.75 1.8125 1.875 1.9375

; Afinacao justa para 1.5 - pelas razões
f9 0 64 -2  12 1.5  1  84  1  1.033333  1.0625  1.1 1.125 1.166666 1.2 1.25 1.3 1.33333 1.4
1.435

; Afinacao Justa em 3.333...(tetracorde lidio nas cromaticas) - pelas razões
f10 0 64 -2 12 3.333333333 1 72 1  1.125  1.25  1.4  1.5  1.66666  1.875  2.133333  2.25
2.5  2.8  3.2

; Afinacao com Harmonicos 77/24 - pelas razões
f11 0 64 -2 12 3.20833333333 1 72 1  1.125  1.25  1.33333  1.5  1.66666  1.75  2  2.25
2.3333  2.66666  3

; Harmonicos do 12 ao 24 - pelo coeficiente para C
f12 0 64 -2 12 2 1.362625 24 12 13 14 15 16 17 18 19 20 21 22 23

; Harmonicos do 16 ao 32
f16 0 64 -2  16 2 2.0439497 36 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

; Harmonicos do 24 ao 48
f24 0 64 -2 24 2 5.4505325 24 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
45 46 47

e
</CsScore>
</CsoundSynthesizer>
```

Algoritmo nº 4

XY / FM

```
;------------------- XY-FM ---------- by: Sergim Veiga --------------
<CsoundSynthesizer>
<CsOptions>
-d --nodisplays -m0
-+rtaudio=JACK ; <-- Cabeçalho com saída para jack 48kHz, stereo e taxa de cotrole 1 por 1.
-+jack_client=XY-FM
-odac:system:playback_
</CsOptions>
<CsInstruments>
sr = 48000
ksmps = 1
nchnls = 2

FLpanel "XY-FM",640,680 ; <----- Abre o painel, determina um título e um tamanho de janela,

; abaixo o opcode FLjoy associa os valores dos eixos X e Y às variáveis gk1 e gk2 declaradas
ali mesmo.

gk1,gk2,ihj1,ihj2 FLjoy "Frequencia portadora = eixo Y // indice modulatorio = eixo X", 1,
256,27.5,880,0,0,-1,-1,600,600,10,10

; Os 4 últimos parâmetros determinam o tamanho do painel (600x600) e a posição na janela
(10x10).

FLrun  ; <----- Fecha o painel e determina abrí-lo.

   instr 1

kporttime linseg 0,0.01,.02,.04,.02
kfreqX   portk gk1,kporttime; <-- Portamento (glissando entre valores - suprime saltos)
kfreqy   portk gk2,kporttime;/

a1 oscili kfreqX*kfreqy, kfreqy, 1 ; <--  MODULADORA --> gk1 * gk2 formam o índice
modulatório.

a2 oscili 4000,kfreqy+a1, 1 ; <-- PORTADORA -->  Frequencia portadora dada por gk2, a mesma
variável da frequência da moduladora.

   outs   a2,a2 ;  <---- Saída do mesmo sinal nos 2 canais.
   endin

</CsInstruments>
<CsScore>
f1 0 4096 10 1  ;  <----  Função senoidal

i1 0 3600   ;   <---- duração de 3600 segundos (1 hora)

e
</CsScore>
</CsoundSynthesizer>
```

Algoritmo nº 5

Saigai to Higai - 災害と被害

```
; ------------------------ Saigai To Higai --------------- Composed by Sergim Veiga, 2014
<CsoundSynthesizer>
<CsOptions>
-d
-odac
;-o/home/sergim/Saigai-to-Higai.wav
;-+rtaudio=JACK
;-+jack_client=Saigai-to-Higai
;-odac:system:playback_
;-odac:Qtractor:Master/in_      ;   <---- conecta direto ao input do Qtractor
</CsOptions>
<CsInstruments>
sr = 44100
ksmps = 128
nchnls = 2
0dbfs = 280000
giosc1 OSCinit 8080
gkx init 0
gky init 0
gkz init 0
gaLinit 0
gaRinit 0
gaterre  init 0
giseno ftgen 1,0,8192,10,1
gienv ftgen 2,0,4096,20,2,1
;========================== ONDAS SÍSMICAS pelo OSC, o acelerometro do andOSC "/acc" ====
instr  55
kans1 OSClisten giosc1, "/acc", "fff", gkx,gky,gkz

   kporttime   linseg0,0.01,.02,.04,.02
   kfreqX        portk   gkx+20, kporttime
   kfreqY        portk   gky+20, kporttime
   kfreqZ        portk   (gkz+10)*2000, kporttime
avib     oscili kfreqZ,kfreqX*kfreqY, giseno
gaterre = avib
endin

instr 1    ;   <--- granulação para os ventos
kfreq linseg p4,p3/2,p5,p3/2,p4
kamp  linseg p6,p3/2,p7,p3/2,p6
kgfreqlinseg p8,p3/2,p9,p3/2,p8
kdeltalinseg p10,p3/2,p11,p3/2,p10
kgrao linseg p12,p3/2,p13,p3/2,p12
a1 grain kamp, kfreq, kgfreq, kamp,kdelta, kgrao, giseno, gienv, giseno
a2 grain kamp, kfreq, kgfreq, kamp,kdelta, kgrao, giseno, gienv, giseno
kenv  linseg 0,.003,1,p3-.006,1,.003,0
outs  a1*kenv,a2*kenv
gaL = gaL + a1
gaR = gaR + a2
;gaterre = gaterre + a2 + a1
endin
;==========================  som de impacto metálico   ===================
instr 2
a1 barmodel    1, 1, p4, 0.001, p5*10, 4, p5, p6, p7
aenv  linseg 0,.003,1,p3-.006,1,.003,0
a2 = a1*aenv
     outs      a2*0,a2*.7
     gaterre = gaterre + a2
;gaL = gaL + (a1*aenv)
;gaR = gaR + (a2*aenv)
endin
;============================================
;========================= som de impacto metálico ====================
instr 7
a1 barmodel    1, 1, p4, 0.001, p5*10, 4, p5, p6, p7
aenv  linseg 0,.003,1,p3-.006,1,.003,0
a2 = a1*aenv
     outs      a2*.7,a2*0
```

```
     gaterre = gaterre + a2
;gaL = gaL + (a1*aenv)
;gaR = gaR + (a2*aenv)
endin
;======================= ruido branco filtrado (água)  ================
instr 3
kamp  linseg p4,p3/2,p5,p3/2,p4
kbeta linseg p6,p3/2,p7,p3/2,p6
a1 noise kamp, kbeta
kenv  linseg 0,.003,1,p3-.006,1,.003,0
outs  a1*kenv, a1*0
gaL = gaL + a1
gaR = gaR + a1
endin
;=======================  senoidais simples para pentatônicas  ==========
instr 4
aRandom = urd (p4)
afreq = aRandom*p5
a1 oscili    3000+gaterre,afreq,giseno
aenv  linseg 0,.02,1,p3-.04,1,.02,0
a2 = a1*aenv
outs a2,a2*0
gaR = gaR + a2
gaL = gaL + a2
endin
;=============================================
instr 5
aRandom = urd (p4)
afreq = aRandom*p5
a1 oscili    3000+gaterre,afreq,giseno
aenv  linseg 0,.02,1,p3-.04,1,.02,0
a2 = a1*aenv
outs a2*0,a2
gaR = gaR + a2
gaL = gaL + a2
endin
;=======================  Senoidais harmônicos de 131 e 137 - Iodo e Césio  ===========
instr 6
kcps  line  p4,p3,p5
ampline  p6,p3,p7
kgama line  p8,p3,p9
kFrGrao  line  p10,p3,p11
k1 randomh 4, kgama, kFrGrao
k2 randomh 4, kgama, kFrGrao
  a1 oscili amp+gaterre,kcps*int(k1), giseno
  a2 oscili amp+gaterre,kcps*int(k2), giseno
  aenv linseg 0,.004,1,p3-.008,1,.004,0
  outs a1*aenv*p12,a2*aenv*p13
gaL = gaL + (a1*aenv*p12)
gaR = gaR + (a2*aenv*p13)
endin
;=============================================
instr  99   ;REVERB

a1 reverb gaL, p4
a2 reverb gaR, p4
  aenv linseg 0,2,p5,p3-4,p5,2,0
outs   a1*aenv,a2*aenv
gaL = 0
gaR = 0
endin
;=======================  EXPLOSAO  ======================
instr 100
kfreq linseg p4/2,p3,p5
kamp  linseg p6,p3,p7
kgfreqlinseg p8,p3,p9
kdeltalinseg p10,p3,p11
```

```
kgrao linseg p12,p3,p13
a1 grain kamp, kfreq, kgfreq, kamp,kdelta, kgrao, giseno, gienv, giseno
a2 grain kamp, kfreq, kgfreq, kamp,kdelta, kgrao, giseno, gienv, giseno
kenv  linseg 0,.003,1,p3-.006,1,.003,0
outs  a1*kenv,a2*kenv
gaL = gaL + a1
gaR = gaR + a2
gaterre = gaterre + ((a1+a2)/4)
endin
</CsInstruments>
<CsScore>;               ----------   PARTITURA    -----------
t 0 50
f3 0 512 -41  1  1  1.125  1  1.265625  1  1.5  1  1.6875  1  2  1  2.25  1  2.53125  1  3
1  3.375  1  4  1  4.5  1  5.0625  1  6  1  6.75  1  8  1
f4 0 512 -41  1  1  1.125  1  1.185185185  1  1.5  1  1.5625  1  2  1  2.25  1  2.37037037
1  3  1  3.125  1  4  1  4.5  1  4.740740741  1  6  1  6.25  1  8  1
;              ----------    acima, funções para pentatônica maior e menor, para os instr 4 e
5
i55 0 272
i99 0 113 4 .17
i99 112 160 22 .2
;==================================== Ondas sismicas ================
;Terremoto      Frequencia   amplitude    graoFreq     Delta     Grao

i1 9     5     22 44    0 4000       11 77    10 60    .03 .03
i1 16 10    22 44    0 8000       11 88    10 120   .03 .03

{ 5 CTR
i1 30 [3+$CTR.]22 44    0 [2000+(~*3000)] 11 77    11 55    .03 .03
i1 +  [5+$CTR.]22 44    0 [2000+(~*3000)] 11 88    11 66    .03 .03
i1 +  [7+$CTR.]22 44    0 [2000+(~*3000)] 11 66    11 77    .03 .03
i1 +  [5+$CTR.]22 44    0 [2000+(~*3000)] 11 88    11 66    .03 .03
i1 +  [5+$CTR.]22 44    0 [2000+(~*3000)] 11 88    11 66    .03 .03
i1 +  [5+$CTR.]22 44    0 [2000+(~*3000)] 11 88    11 66    .03 .03
}

i100   144     12    55 0     3000 0   200 0    300 0    .03 .03
i100   .    .     73 0      .   0 .   .    .   . .03 .03
i100   .    .     97 0      .   0 .   .    .   . .03 .03
i100   .    .     130 0     .   0 .   .    .   . .03 .03
i100   .    .     174 0     .   0 .   .    .   .  .03 .03
i100   .    .     231 0     .   0 .   .    .   . .03 .03
i100   .    .     309 0     .   0 .   .    .   . .03 .03
i100   .    .     412 0     .   0 .   .    .   . .03 .03
i100   .    .     549 0     .   0 .   .    .   . .03 .03
i100   .    .     732 0     .   0 .   .    .   . .03 .03
i100   .    .     976 0     .   0 .   .    .   . .03 .03
i100   .    .     1302 0    .   0 .   .    .   . .03 .03

i1 17210    22 44    0 3000       11 77    10 60    .03 .03
i1 +  1     22 44    0 6500       11 77    10 60    .03 .03
i1 +  3     22 44    0 4000       11 77    10 60    .03 .03
i1 +  6     22 44    0 2000       11 77    10 60    .03 .03
i1 +  8     22 44    0 5000       11 77    10 60    .03 .03
i1 +  7     22 44    0 4000       11 77    10 60    .03 .03
i1 +  6     22 44    0 2000       11 77    10 60    .03 .03
i1 +  5     22 44    0 8000       11 77    10 60    .03 .03
i1 +  4     22 44    0 3600       11 77    10 60    .03 .03
i1 +  3     22 44    0 4000       11 77    10 60    .03 .03
i1 +  2     22 44    0 5000       11 77    10 60    .03 .03
i1 +  1     22 44    0 6500       11 77    10 60    .03 .03
i1 +  13    22 44    0 0          11 77    10 60    .03 .03
i1 +  2     22 44    0 2000       11 77    10 60    .03 .03
;==================================== Barras de metal============
;         Tamanho      Posição      Amplitude Espacial
{ 5 CT
i2 [33+($CT. * .2)] [.6+(~*2)]  [10+(~*10)] [.1+(~*.7)] [2200+(~*600)] 0.01
```

```
{ 9 CM
i2 + [($CM.*.2)+(~*2)]  [10+(~*10)] [.1+(~*.7)] [1500+(~*900)]  .01
i2 + [($CM.*.2)+(~*2)]  [10+(~*10)] [.1+(~*.7)] [1500+(~*900)]  .
}
}

{ 5 CT
i7 [33+($CT. * .2)] [.6+(~*2)]  [10+(~*10)] [.1+(~*.7)] [2200+(~*600)] 0.01
{ 9 CM
i7 + [($CM.*.2)+(~*2)]  [10+(~*10)] [.1+(~*.7)] [1500+(~*900)]  .01
i7 + [($CM.*.2)+(~*2)]  [10+(~*10)] [.1+(~*.7)] [1500+(~*900)]  .
}
}


{ 5 CT
i2 [120+($CT. * .2)] [.6+(~*2)]  [10+(~*10)] [.1+(~*.7)] [1000+(~*600)] 0.01
{ 7 CM
i2 + [($CM.*.2)+(~*2)]  [(10-$CM)+(~*10)] [.1+(~*.7)] [1000+(~*900)]  .01
i2 + [($CM.*.2)+(~*2)]  [(10-$CM)+(~*10)] [.1+(~*.7)] [1000+(~*900)]  .
}
}

{ 5 CT
i7 [120+($CT. * .2)] [.6+(~*2)]  [10+(~*10)] [.1+(~*.7)] [1000+(~*600)] 0.01
{ 7 CM
i7 + [($CM.*.2)+(~*2)]  [(10-$CM)+(~*10)] [.1+(~*.7)] [1000+(~*900)]  .01
i7 + [($CM.*.2)+(~*2)]  [(10-$CM)+(~*10)] [.1+(~*.7)] [1000+(~*900)]  .
}
}
;============================== Vento e Agua===================
;Vento      Frequencia   amplitude graoFreq  Delta     Grao
{ 3
i1 70 [2+(~*5)]27.5 55     0 1000   33 77    20 60    .05 .02
{ 3
i1 +  [2+(~*5)]110 55   0 2000   33 77    20 100   .05 .02
i1 +  [2+(~*5)]220 55   0 3000   33 77    20 200   .05 .02
i1 +  [2+(~*5)]330 55   0 4000   33 77    20 400   .05 .02
i1 +  [2+(~*5)]330 55   0 3000   33 77    20 400   .05 .02
}
}
i1 12832 33 880   3000 3000   33 77    20 400   .02 .02

;Agua    Volume   Filtro
{ 3
i3 80 [2+(~*5)]3000 30001  .999
{ 3
i3 + [3+(~*4)] 2000 30001  .998
i3 + [4+(~*4)] 2000 50001  .994
i3 + [5+(~*4)] 2000 40001  .96
}
i3 + [18+(~*5)]400 5000    1  .5
}
;====================== música local ====================
;           escala Freq
i4 1 [(~ * .7) + .02 ]     3  220
{ 50 CT
i4 + [(~ * .7) + .02 ]     3  220
}
{ 130 CT
i4 + [(~ * .7) + .02 ]     4  220
}

i5 1 [(~ * .7) + .02 ]     3  220
{ 50
i5 + [(~ * .7) + .02 ]     3  220
}
```

```
{ 130
i5 + [(~ * .7) + .02 ]      4  130.812782
}
;======================= Radiação ==================
;      Freq         Amp       Gama      grainFreq
i6 118 26.001 27.5   100 3000    512 512     200 200     1 1
;Césio
i6 14416 137 137     3000 800 512 137     200 25   1 0
i6 160 90   137 137     800 100     137 24   25 9     1 0
;Iodo
i6 14416 131 131     3000 800 512 131     200 24   0 1
i6 160 90   131 131     800 100     131 24   24 7     0 1
e
</CsScore>
</CsoundSynthesizer>
```

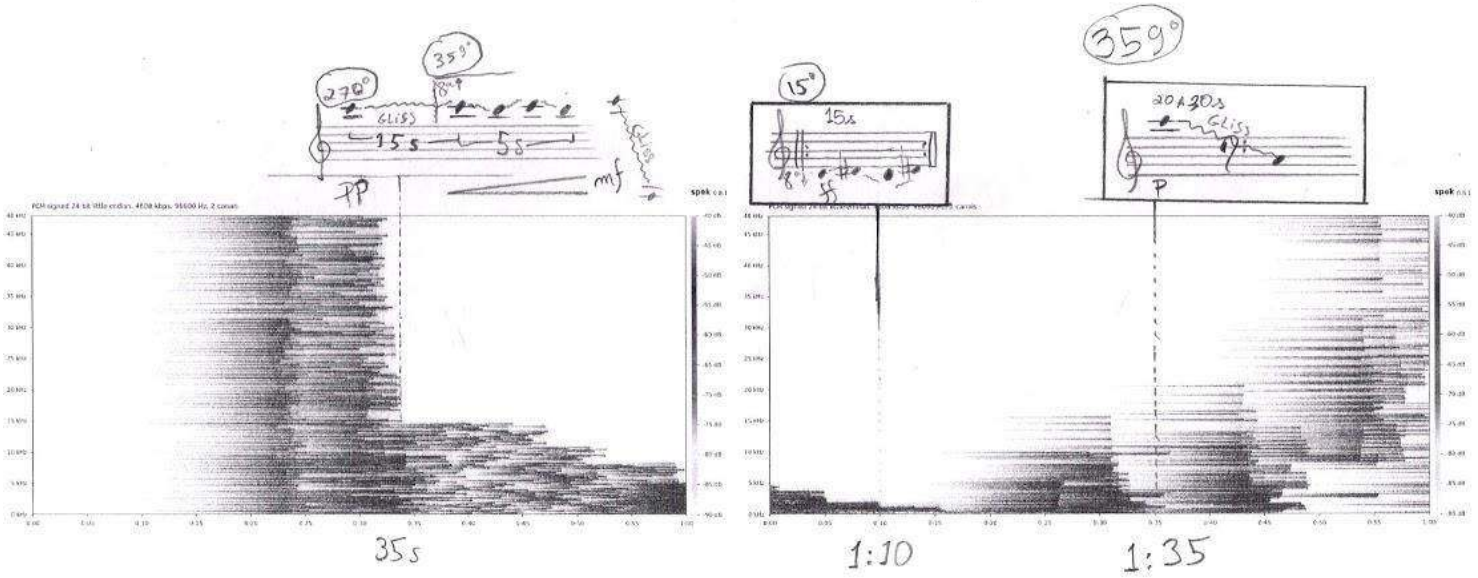
ANEXO B

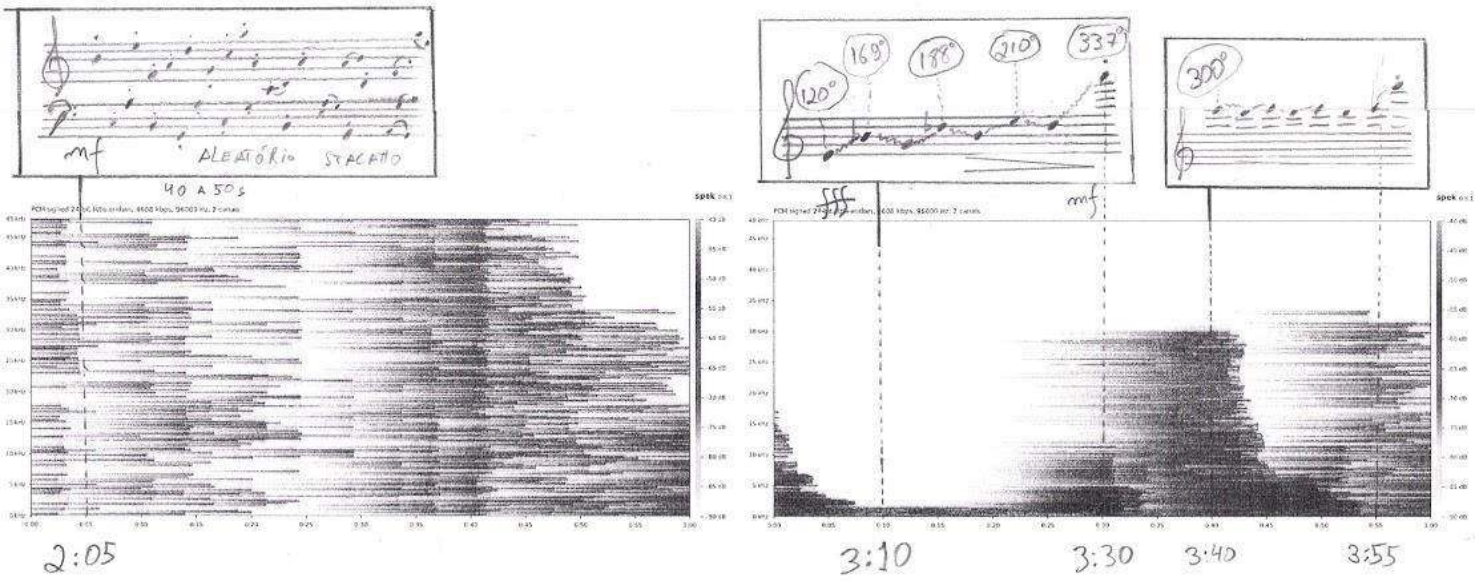
Partituras

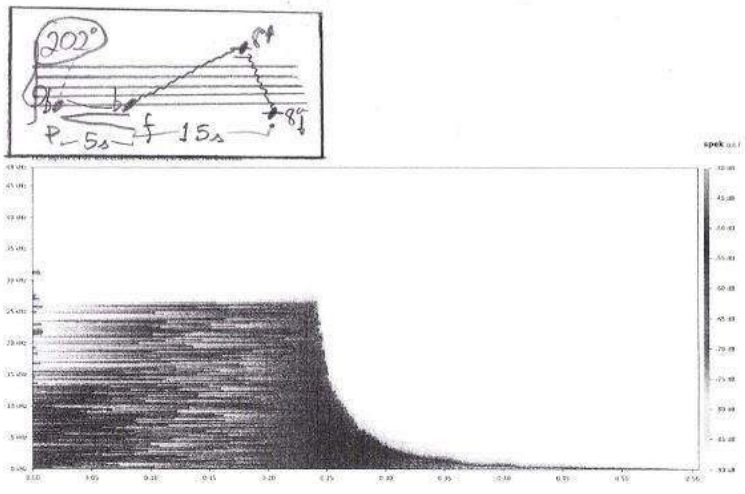
Vênus

Para Theremin Giroscópico e
Síntese por Randomização de
Harmônicos

Sergim Veiga



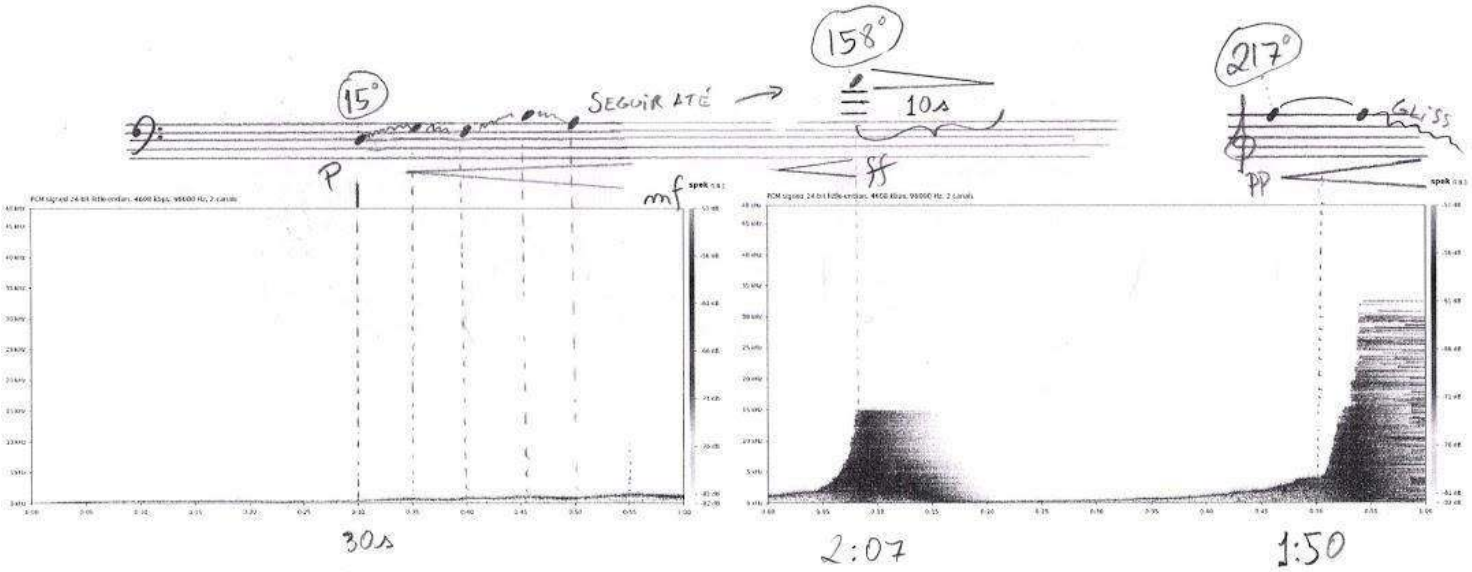


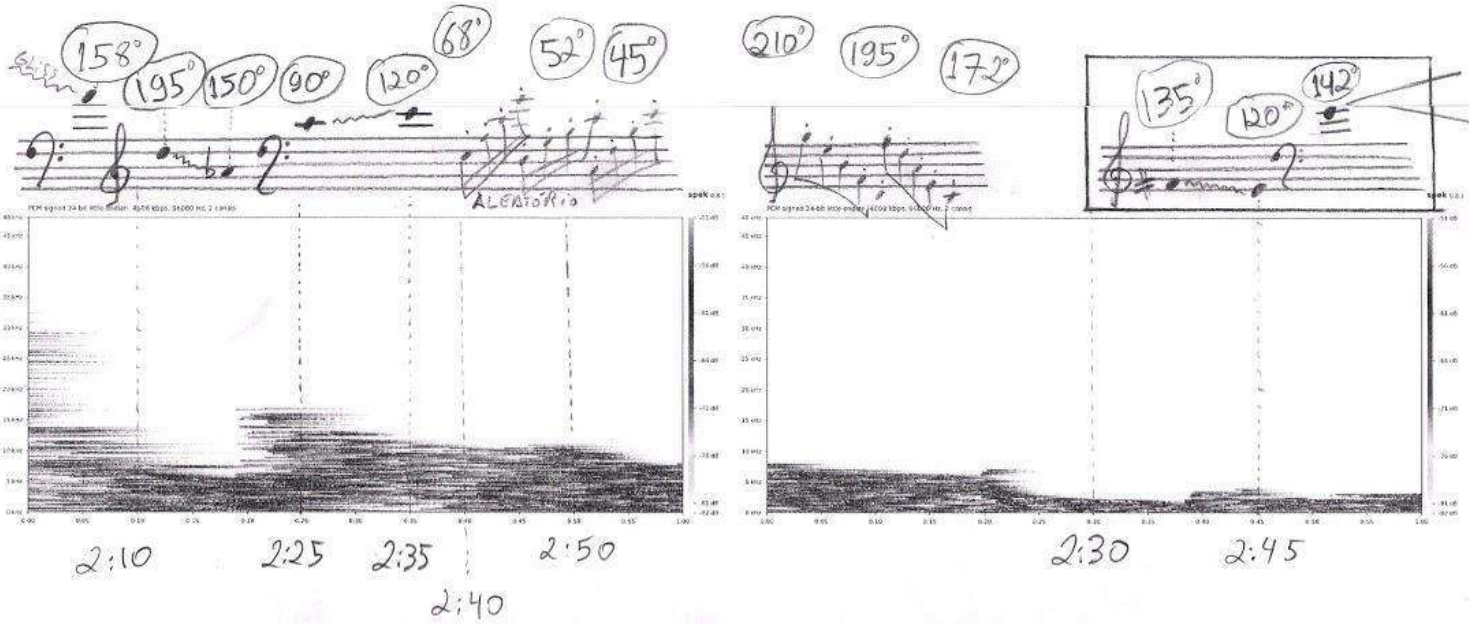


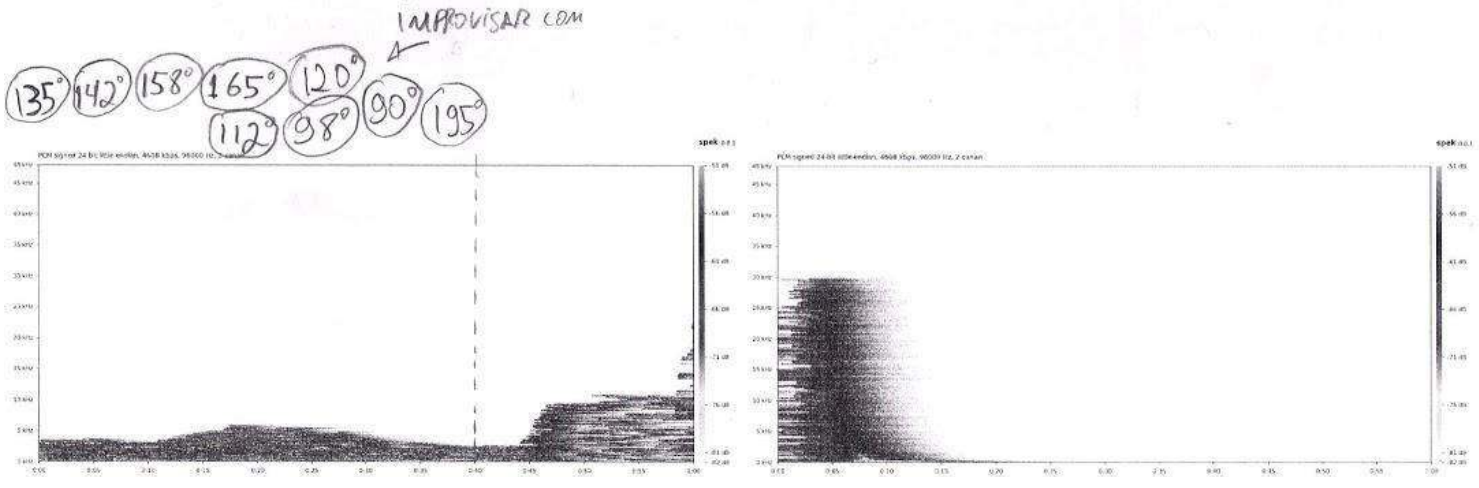
Marte

Para Theremin Giroscópico e
Síntese por Randomização de
Harmônicos

Sergim Veiga

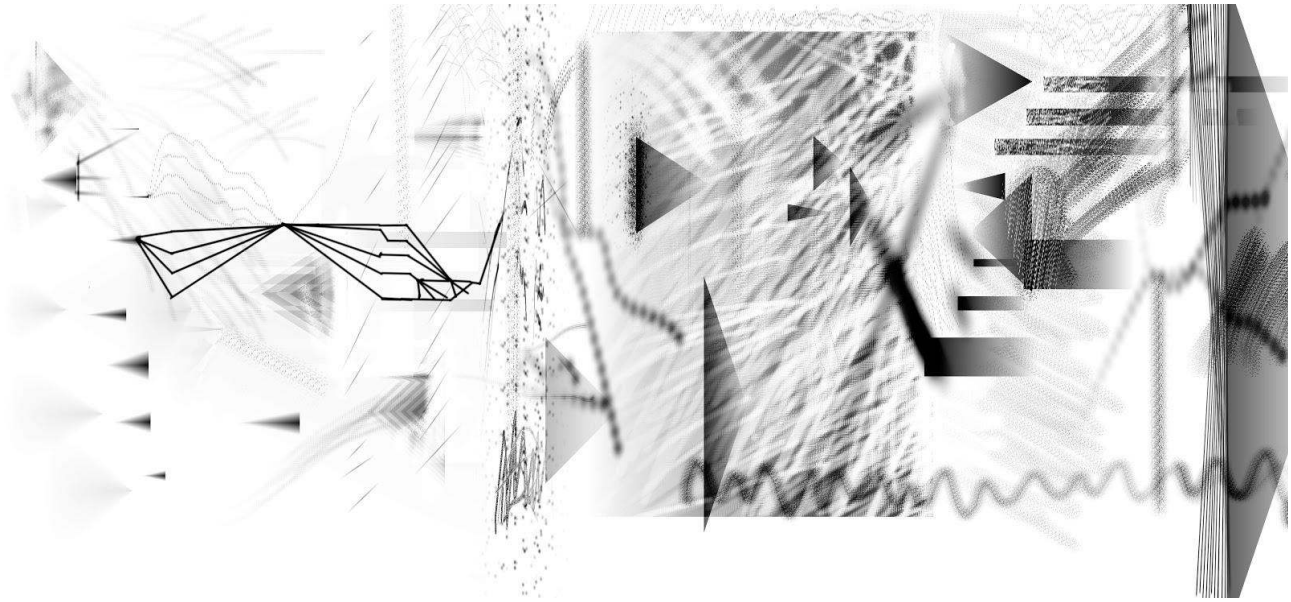


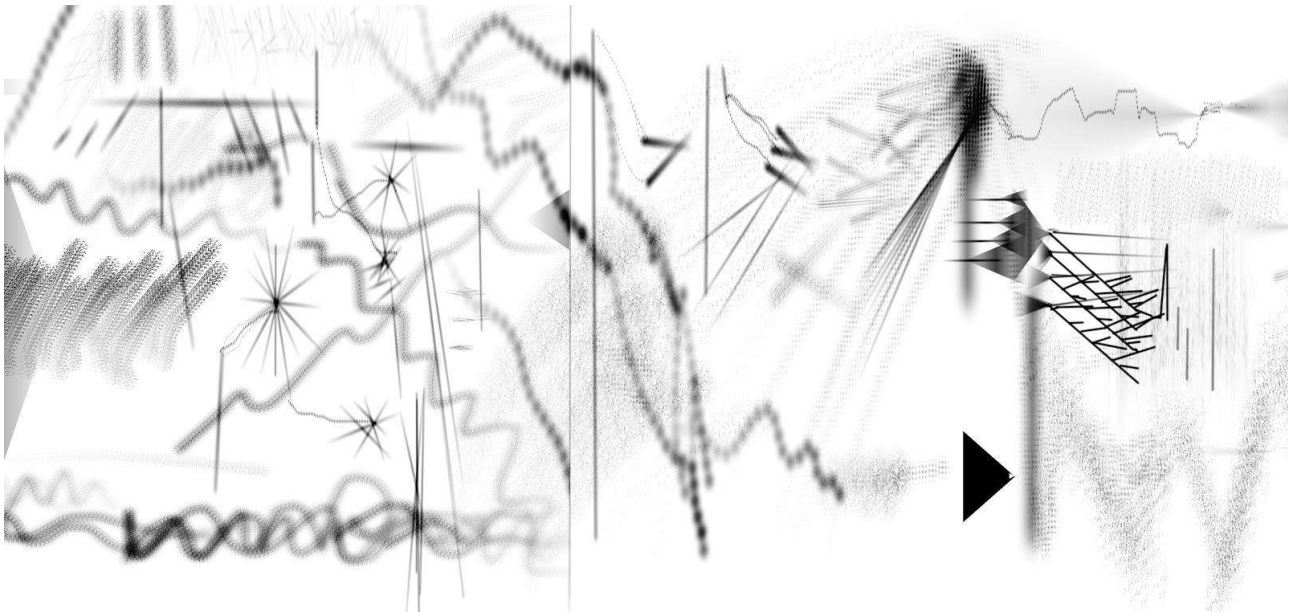


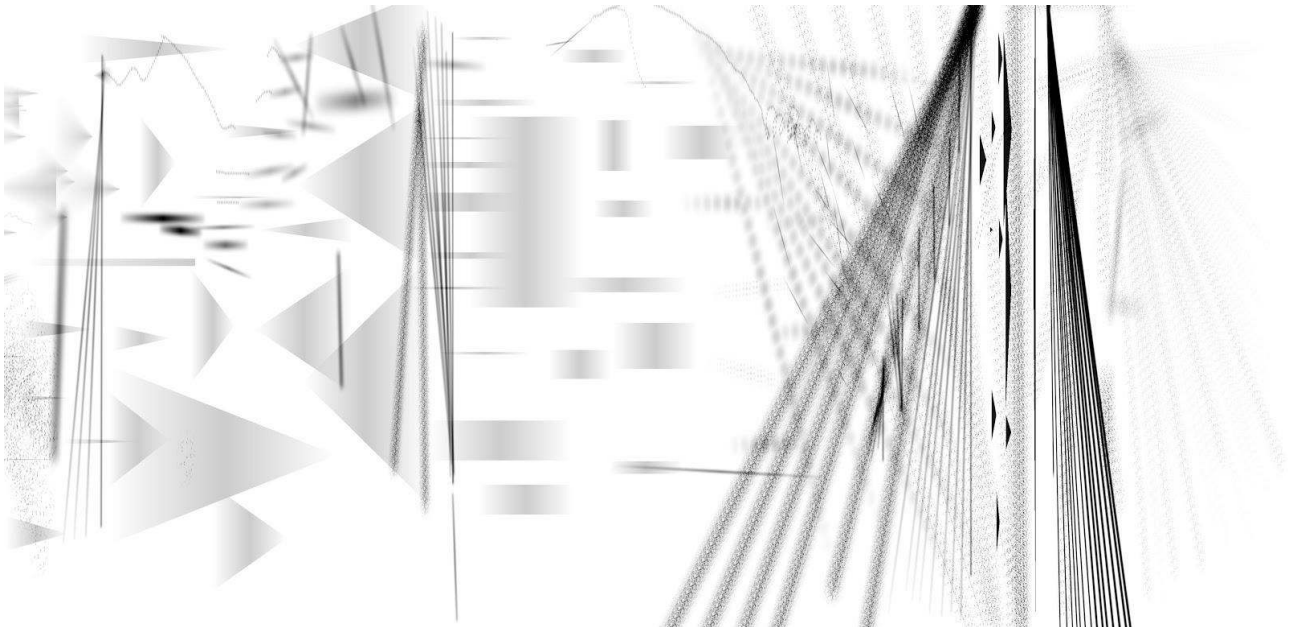


ANS Estudo Nº1

Sergim Veiga (2015)

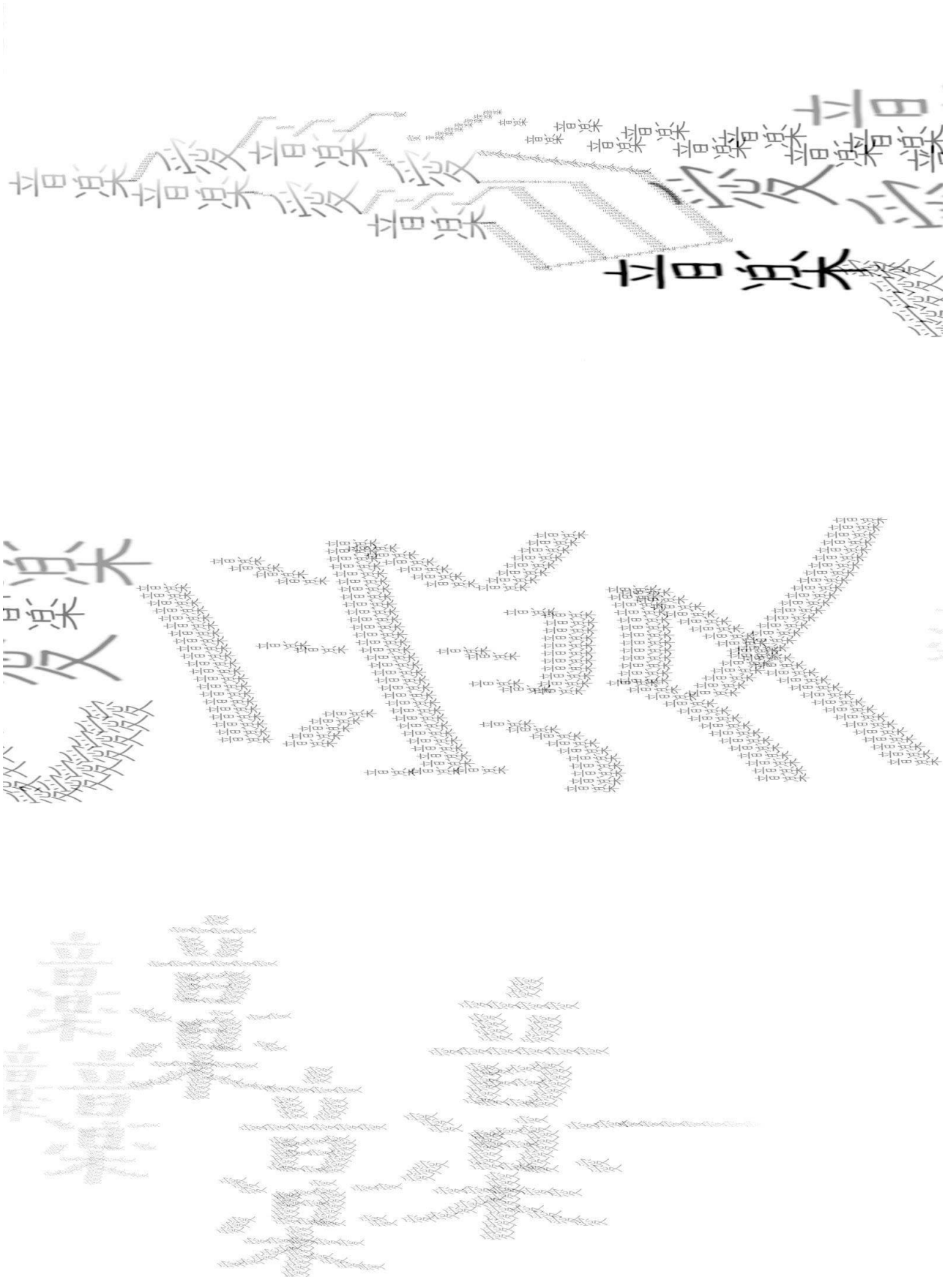






ANS Estudo Nº2

Sergim Veiga (2015)



ANEXO C

Lista de *Softwares* Livres

1. **UBUNTU STUDIO** - http://cdimage.ubuntu.com/ubuntustudio/releases/

O Ubuntu Studio reúne os mais avançados *softwares* Livres voltados à produção musical e modelagem sonora como: editores de partitura, editores muiti-pista, sintetizadores, *Csound*, *Puredata*, etc. Utiliza um kernel LINUX preemptivo, especial para baixa latência e essencial para síntese sonora em tempo real.

2. **ALSA** - http://www.alsa-project.org/main/index.php/Main_Page

O *Advanced Linux Sound Architecture* (ALSA) fornece a funcionalidade de áudio e MIDI para o sistema operacional Linux e sua característica mais importante é ter um suporte eficiente para todos os tipos de placas de som e interfaces de áudio multicanal.

3. **JACK CONTROL** - http://jackaudio.org

O *Jack Control* é um servidor de áudio para performance em tempo real, equivalente ao *driver* ASIO encontrado em sistemas WINDOWS ou COREAUDIO MAC. Ele é responsável por reduzir ao máximo o tempo de latência do processamento do sinal.

4. **QTRACTOR** - http://qtractor.sourceforge.net/qtractor-index.html

O *Qtractor* é um sequenciador multi-pista de áudio ou MIDI escrito em C ++, sua plataforma de destino é o Linux, onde o *Jack Audio Connection Kit* (Jack) para o áudio, e o *Linux Advanced Sound Architecture* (ALSA) para o MIDI, são suas principais infra-estruturas. Ele é especialmente dedicado ao *home-studio* pessoal.

5. **QUTECSOUND** - http://qutecsound.sourceforge.net/

O *QuteCsound* está em sua versão 0.8.3. que inclui muitas correções e alterações, em especial, a adição deste *front-end* como depurador padrão que acompanha o *Csound* 6.03 lançado recentemente, inclui também, capacidades de depuração.

6. **SPEK** - http://spek.cc/

Spek (Tradução: “o bacon” em holandês) ajuda a analisar seus arquivos de áudio, mostrando seu espectrograma. *Spek* é um *Software* Livre disponível para UNIX, WINDOWS e MAC OS X. Licenciado sob a GPLv3. O projeto é escrito em C ++, o código está disponível no GitHub.

7. **GIMP** - http://www.gimp.org/

GIMP é um acrônimo para GNU *Image Manipulation Program*. É um programa livremente distribuído para tarefas como o retoque de fotos e composição de imagem. Popularmente conhecido como o *Photoshop* do Linux. É escrito e desenvolvido sob X11 em plataformas UNIX, mas, é basicamente o mesmo código executado em MS Windows e Mac OS X.

8. **VIRTUAL ANS** - http://www.warmplace.ru/soft/ans/

O *VirtualANS* é um simulador do ANS, o único emulador desse sintetizador russo - instrumento microtonal fotoelétrico/espectral musical criado pelo engenheiro russo Evgeny Murzin de 1938 a 1958. Murzin nomeou sua invenção em homenagem ao compositor Alexander Nikolayevich Scriabin. O criador do *software* é Alexander Zolotov que distribui gratuitamente o programa para computadores.